A potential flow approach to complex 3D flows with actuation surfaces

S.R. Gamme

December 21, 2015
A potential flow approach to complex 3D flows with actuation surfaces

MASTER OF SCIENCE THESIS

For obtaining the degree of Master of Science in Engineering Wind Energy at Technical University of Denmark and in Aerospace Engineering at Delft University of Technology.

S.R. Gamme

December 21, 2015
European Wind Energy Master - EWEM
Of
Rotor Design

The undersigned hereby certify that they have read and recommend to the European Wind Energy Master - EWEM for acceptance a thesis entitled “A potential flow approach to complex 3D flows with actuation surfaces” by S.R. Gamme in partial fulfillment of the requirements for the degree of Master of Science.

Dated: December 21, 2015

Head of department: prof.dr. G.J.W. van Bussel
Delft University of Technology, the Netherlands

Supervisor: ir. G.L. de Oliveira Andrade
Delft University of Technology, the Netherlands

Supervisor: dr. ir. N. Ramos Garcia
Danish Technical University, Denmark

Reader: prof.dr.ir. L.L.M. Veldhuis
Delft University of Technology, the Netherlands
Potential flow methods originated in 1910 when Joukowski [1] represented a 2D lifting flow by a single load vortex. Even though algorithms were developed to analyse 3D flow fields, the non-existing computational power limited their use. With the advent of computational power in the 1960s, potential panel codes appeared and became standard tools in aerodynamic design.

The rise of computational power opened opportunities to model viscosity and turbulence as the Navier-Stokes equations could be discretized on a grid. Since all phenomena in the fluid dynamics is explicitly taken into account, these methods are often preferred with respect to panel methods. However, CFD codes typically diffuse vorticity, requiring a very fine meshing to conserve all vortical structures. This limits the use of CFD on actuation surfaces in terms of accuracy and numerical time. Panel methods explicitly enforce this vorticity conservation and can be preferred in wake problems. Furthermore the number of unknowns is limited to the number of singularities on the geometry. However, for very fine meshing, the use of panel codes might still be prohibitive. This is due to the scaling of the computational time with $O(N^2)$, with $N$ being the number of panels.

Fortunately, in the field of mathematics a lot of effort has been devoted to the rapid evaluation of potential fields which involve a large number of particles which interact with each other. The Fast Multipole Method (FMM) can reduce the traditional scaling with $O(N^2)$ to $O(N)$. This however would not alleviate the numerical cost associated with setting up a linear set of equations since the isolated influence of each panel on each collocation point is needed. Therefore the FMM can be coupled with an iterative solver which can bypass the requirement of explicitly setting up a linear set of equations, known as FastBEM methods.

These Fast algorithms were combined with a panel code to reduce the traditional quadratic scaling to a linear scaling with the objective to simulate inviscid complex 3D flows with actuation devices. Particularly successful is the iterative wake expansion behind an actuator disk, which accelerated by a factor 30 to a precision of $10^{-9}$, for 60000 panels. The code can also deal with lifting surfaces and proved accurate when compared to Prandtl’s lifting line theory. Finally an actuator was modelled within a wind tunnel and good agreement was found for the wall effects between the developed Fast panel code and CFD results.
Acknowledgements

I would like to express my gratitude to my supervisors for their moral and academic support throughout this thesis work. Thank you Nestor for helping me to set up this project and advising me, you were always available when I faced some problems.

Gael, without you this project would never have reached this stage. I appreciate the countless hours you spent to help me solve the issues I was facing. The whole weekend you spent on figuring out the problem in the FMM source code is an excellent example of your commitment to the project.

Next to the supervisors, I also would like to thank the committee members. Thanks for showing your interest and freeing the time to assess this work.

Thanks to all my friends that not only morally supported me, but also reminded me sometimes there is more in life than this thesis project.

Finally I would like to thank my parents and my sister for their unconditional support. Not only during the thesis work, but the whole 5 years of my higher education they where there for me.
# Contents

Summary ............................................................... v

Acknowledgements .................................................. v

**Introduction** ..................................................... 1

1 Motivation and objective ........................................... 1

2 Research questions ................................................ 3

3 Research plan ........................................................ 3

4 Report outline ....................................................... 4

1 Potential flow solutions ............................................ 5

1.1 Helmholtz decomposition ......................................... 5

1.2 Laplace equation .................................................. 6

1.3 Vorticity and circulation .......................................... 7

1.4 Singularity elements .............................................. 8

1.4.1 Point source ................................................... 8

1.4.2 Point dipole ................................................... 9

1.4.3 Point vortex .................................................. 10

1.5 Kutta condition ................................................... 11

2 A 3D panel code ..................................................... 13

2.1 Singularity elements .............................................. 13

2.1.1 Source panel ................................................ 14
2.1.2 Dipole panels ................................................. 14
2.1.3 Vortex rings .................................................. 15
2.2 Linear set of equations ....................................... 16
  2.2.1 Impermeability condition ................................. 16
  2.2.2 Kutta condition ............................................ 17
2.3 Iterative wake method ...................................... 18
  2.3.1 Wake alignment ............................................. 18
  2.3.2 Convergence parameters ................................. 19
2.4 Secondary calculations .................................... 20
  2.4.1 On-surface velocities ................................... 20
  2.4.2 Force coefficients ....................................... 20
2.5 Implementation .............................................. 21
  2.5.1 Presolver .................................................... 22
  2.5.2 Solver ....................................................... 24
  2.5.3 Post-processor ............................................ 24

3 Fast algorithms .............................................. 27
  3.1 Fast Multipole Method .................................... 27
    3.1.1 Theory .................................................... 28
    3.1.2 Method .................................................... 29
    3.1.3 Results .................................................. 30
  3.2 Iterative solvers .......................................... 32
    3.2.1 Preconditioning ....................................... 33
    3.2.2 Jacobi .................................................... 34
    3.2.3 GMRES ..................................................... 34
    3.2.4 Results .................................................. 36
  3.3 FastBEM ...................................................... 39
    3.3.1 FMM and Jacobi ....................................... 39
    3.3.2 FMM and GMRES ...................................... 40

4 Fast Panel Code .............................................. 43
  4.1 Methods .................................................... 43
  4.2 Numerical study ........................................... 45
    4.2.1 Scaling factors ....................................... 45
    4.2.2 Case study .............................................. 46
  4.3 Outcome ..................................................... 47
5 Verification and Validation

5.1 The actuator disk

5.1.1 Wake convergence

5.1.2 Verification using 1-D momentum theory

5.1.3 Verification of mass convergence

5.1.4 Time analysis

5.2 Impermeable wall

5.2.1 Analytical solution

5.2.2 Numerical results

5.3 The 3D wing

5.3.1 Elliptical load distribution

5.3.2 Wake roll-up

5.3.3 Time analysis

5.4 Actuator in a wind tunnel

5.4.1 Method

5.4.2 Velocity at disk

5.4.3 Power coefficient

6 Future developments

6.1 Envisioned applications

6.2 Perspective for future use

6.2.1 Boundary layer coupling

6.2.2 Aeroacoustic noise

6.2.3 Static aeroelastic analysis

7 Conclusions and recommendations

7.1 Conclusions

7.2 Recommendations

Appendices

A $C_T - \omega_T$ relation for an actuator disk
Introduction

1 Motivation and objective

Over a century ago, Joukowski [1] represented a 2D lifting flow by a single load vortex which marked the start of potential flow methods. This was extended to 3D by Prandtl [2] and became known as the lifting line method. Rosenhead developed a method to discretize a vortex sheet with elemental vortices of constant strength [3]. However, due to the nonexisting computational power, the potential of these methods only became apparent in the 1960s. Then Hess and Smith [4] distributed singularities on a 2D airfoil surface to investigate the flow around a body. Later this was extended to 3D and it formed a basis for commercial panel codes such as PANAIR and VSAERO. Such codes are still widely used for initial aerodynamic design of aircraft, road vehicles and other applications. Especially for fully attached, high Reynolds numbers, subsonic flows these are valuable tools.

The rise of computational power opened opportunities to model the effect of viscosity and turbulence as the Navier-Stokes equations could be discretized explicitly on a grid. In Computational Fluid Dynamics (CFD), this discretization was initially achieved by applying a finite difference method, and with the increasing computing capabilities also finite volume and finite element methods appeared. The advantage w.r.t. panel codes is that phenomena in the fluid dynamics are explicitly taken into account. However, these methods typically diffuse vorticity, requiring a very fine meshing to conserve all vortical structures.

In panel codes vorticity conservation is explicitly enforced. This makes them suitable for problems with large and complex wakes geometries, such as wake interaction problems or wind turbines. Furthermore the number of unknowns can be restricted to the the number of singularities distributed on the objects in the flow. In CFD the unknowns for each fluid cell are to be solved, significantly increasing computational time.

Even though panel methods use less elements, the numerical time can still be prohibitive for large panel counts. If one wants to solve for $N$ unknown singularity strengths, the influence of all singularities on each singularity must be known. This requires $O(N^2)$
Introduction

operations, meaning computational times scale with $N^2$. In other words, doubling the panel counts, squares the computational time.

Fortunately in the field of mathematics, a lot of effort has been devoted to the rapid evaluation of potential and force fields, which involve a large number of particles whose interactions are Coulombic or Gravitational in nature. In particular the Fast Multipole Method (FMM) proposed by Greengard and Rokhlin [5] can be used for a rapid solution to these potential problems. This method reduces the traditional scaling with $O(N^2)$ to $O(N)$. This algorithm originated in the 1980s and became one of the most important algorithms developed in the 20th century. It was applied in amongst others the fields of celestial mechanics, finance, statistics, acoustics, chemistry and materials engineering. The FMM even gained more popularity when it was coupled with iterative solvers, as done by Peirce and Napier [6] for elastostatic problems. This allows to solve the potential field without setting up a linear set of equations explicitly. This sped up the traditional Boundary Element Method (BEM), hence receiving the name FastBEM [7]. This method allowed for the modelling of large-scale problems on desktop computers.

Various panel methods have been developed in the past to compute the three-dimensional flow field around wings and wakes. Traditionally, these are not scalable, meaning a careful trade-off between accuracy and computational time has to be made when using them. Even though the described Fast algorithms can be applied on panel methods, no readily available panel code incorporates these features. Combining the FMM and Fast BEM with the flow solver would allow scalability. Another advantage of the FMM is the user-defined precision of the solution, assuring sufficiently accurate solutions with a significant time benefit. A well-documented open source Fast panel code would therefore serve well to the scientific community.

Furthermore it can be the basis for a novel open-source framework of potential code modules. Modules can be added for viscous corrections, aeroacoustics, static aeroelastic analysis and so on. The first building block is developed in this thesis work as a proof of concept, to show the benefits of using Fast algorithms in panel codes.

Thanks to its improved numerical efficiency, a large amount of panels can be employed. This is an essential property in the analysis of propellers and wind turbines which relies on the wake geometry (far) downstream. Standalone actuator disks do not explicitly satisfy an impermeability boundary condition. However, they are often combined with surfaces which do require this explicit boundary condition. Typical examples are: propellers near a wing, shrouded rotors, rotors in a wind tunnel and so on. So next to being numerical efficient, the code should be flexible to handle a wide variety of cases.

Hence it is clear that a well-documented Fast, flexible and open-source panel code can be a valuable tool in many aerodynamic applications. With this in mind the objective of the thesis work can be formulated:

**Objective of the Master thesis research:**

Develop a computationally efficient, accurate and robust implementation of a Fast inviscid panel code to model complex 3D flows with actuation surfaces
2 Research questions

In order to reach the objective, three main research questions are composed:

1. What is the best approach to model complex 3D flows with actuation surfaces?
2. Is an FMM, combined with a FastBEM method effective in reducing the computational time compared to a traditional inviscid iterative panel method?
3. When is it beneficial to use a Fast panel code?

The first question relates to different types of flows and will therefore be answered by analysing several validation cases. These should shed light on the following subquestions:

1. Is an iterative wake procedure adequate to model actuation surfaces such as propellers and wind turbines?
2. Can an explicit impermeable boundary condition be applied on a vortex sheet to represent solid walls?
3. Is the code able to solve the flow field around lifting surfaces when enclosing the Kutta condition?

3 Research plan

To answer the research questions defined before, different tasks were identified:

1. Obtain knowledge about the theory of traditional panel codes and implement a basic 3D iterative wake solver.
2. Study the FMM algorithm and couple the FMM library to the developed solver.
3. Explore the FastBEM method to further reduce the computational time.
4. Assess the computational effort compared to the traditional panel codes
5. Validate the Fast panel code for relevant flow cases found in literature

The code will be validated against aerodynamic applications with increasing level of complexity to gradually verify the working of the code. Furthermore they should employ a large amount of panels to demonstrate the potential of the Fast panel code.

1. The actuator disk
   Since no explicit boundary condition is to be satisfied here, the iterative wake procedure can be verified through the wake expansion/contraction due to the actuator.
2. An impermeable wall
Now an explicit impermeable boundary condition is added to a vortex sheet to model a nozzle. The velocity field can be compared to the theoretical one to ensure impermeability at the surface.

3. The standalone wing
   A 3D lifting wing can be used to check the Kutta condition and on-surface velocity calculation. Analytical circulation values and force coefficients can be used as a reference.

4. The actuator in a wind tunnel
   A combination of the actuator and an impermeable wall is implemented and compared to reference data to show the potential of the code to analyse complex flow cases efficiently.

4 Report outline

The first chapter is an introduction to potential flow, the governing equations and the fundamental solutions to it. These are used in the next chapter to develop a standard 3D panel. It explains how one can arrive at a linear set of equations and solve for the singularity strengths. Also some practical considerations of the code are given. Chapter 3 explains the Fast algorithms which could be used to eventually reduce the standard quadratic scaling to a linear one. Next to the theory, also some numerical experiments were carried out to show their potential for the Fast panel code. In Chapter 4, the Fast algorithms are combined to the developed panel code and an estimation of the numerical efficiency will be done. Then validation cases are treated in Chapter 5. Then the potential role of this code is treated in Chapter 6, including envisioned applications and possible further extensions. Finally some conclusions can be drawn and recommendations can be given in Chapter 7.
Chapter 1

Potential flow solutions

Due to the complex nature of fluid dynamics simplifications are often made. An example of this is potential flow, where the flow is assumed to be inviscid, irrotational and incompressible. In this chapter it will be highlighted how a flow field can be represented with those assumptions. First Helmholtz decomposition is presented to introduce flow potential. Then the Laplace equation for the scalar potential is set up with a solution strategy using Green’s function. This is followed by an introduction on the concepts of vorticity and circulation. Then fundamental flow solutions to the Laplace equation will be given, which will be the building elements for the panel code. Finally some remarks are given on the Kutta condition which has to be satisfied to have a physical potential flow solution.

1.1 Helmholtz decomposition

The 3D velocity field is a function of 3D space \((x, y, z)\) into 3 variable velocity components \((u, v, w)\):

\[ V : (x, y, z) \in \mathbb{R}^3 \rightarrow (u, v, w) \in \mathbb{R}^3 \tag{1.1} \]

Helmholtz theorem [8] dictates that this vector field can be decomposed as the sum of the gradient of a scalar potential \(\phi\) and the curl of a vector potential \(\psi\):

\[ V = \nabla \phi - \nabla \times \psi \tag{1.2} \]

This scalar potential is a function of 3D space \(x, y, z\) into a single variable whereas the vector potential is a function of 3D space into 3 variables. Thus:

\[ \phi : (x, y, z) \in \mathbb{R}^3 \rightarrow \phi \in \mathbb{R} \]
\[ \psi : (x, y, z) \in \mathbb{R}^3 \rightarrow (\psi_x, \psi_y, \psi_z) \in \mathbb{R}^3 \tag{1.3} \]
Consequently the separate terms of the Helmholtz decomposition can be assigned to a coordinate space:

$$\nabla \phi : (x, y, z) \in \mathbb{R}^3 \rightarrow \left( \frac{\partial \phi}{\partial x}, \frac{\partial \phi}{\partial y}, \frac{\partial \phi}{\partial z} \right) \in \mathbb{R}^3$$

$$\nabla \times \psi : (x, y, z) \in \mathbb{R}^3 \rightarrow \left( \frac{\partial \psi_x}{\partial y} - \frac{\partial \psi_y}{\partial z}, \frac{\partial \psi_y}{\partial x} - \frac{\partial \psi_z}{\partial y}, \frac{\partial \psi_z}{\partial x} - \frac{\partial \psi_x}{\partial y} \right) \in \mathbb{R}^3$$

(1.4)

### 1.2 Laplace equation

In potential flow, also the continuity condition must be satisfied which dictates that the net mass flux to/from the system is zero. For incompressible flows the continuity equation reads:

$$\frac{\partial u}{\partial x} + \frac{\partial v}{\partial y} + \frac{\partial w}{\partial z} = 0 \quad \Leftrightarrow \quad \nabla \cdot \mathbf{V} = 0 \quad (1.5)$$

If there would be sources or sinks in the flow, injecting or absorbing mass at point \( x = (x, y, z) \) with rate \( \rho \Phi \), the right hand side would become \( q_x \). Nevertheless when substituting \( \mathbf{V} \) in Equation 1.5 with Equation 1.2, it can be shown that:

$$\nabla \cdot \mathbf{V} = \nabla \cdot (\nabla \cdot \phi) - \nabla \cdot (\nabla \times \psi) = \nabla \cdot (\nabla \cdot \phi) = \nabla^2 \phi = 0 \quad (1.6)$$

Which is the homogeneous form of the Poisson equation, being the Laplace equation. This is a second order linear differential equation which can be solved by Green’s approach. The solution of the equation is the potential \( \phi \) and is obtained as the convolution \( G \ast q \) of the Green’s function \( G \) with the forcing function \( q \) over the whole domain:

$$\nabla^2 (\phi(x)) = q(x) \quad \Leftrightarrow \quad \phi(x) = G(x, \tilde{x}) \ast q(\tilde{x}) = \int_\Omega G(x, \tilde{x}) q(\tilde{x}) d\tilde{x} \quad (1.7)$$

Here \( x = (x, y, z) \) denotes the point at which the potential is sought and \( \tilde{x}(\tilde{x}, \tilde{y}, \tilde{z}) \) are the points over which the integration is performed. For the 2D case the integration domain \( \Omega \) is \( \mathbb{R}^2 \), whereas 3D would require a volume integration over \( \mathbb{R}^3 \). The Green’s functions are known from literature [9] and for the Poisson/Laplace equation it reads:

$$G^{2d}_{(x, \tilde{x})} = \frac{1}{2\pi} \ln \left| x - \tilde{x} \right| \quad , \quad G^{3d}_{(x, \tilde{x})} = \frac{1}{4\pi} \frac{1}{\left| x - \tilde{x} \right|} \quad (1.8)$$

So depending on the forcing term \( q \), different fundamental solutions can be generated such as sources and sinks. Due to the linearity of the Laplace equation, a superposition of these solutions is also a solution. A dipole for example is a source and a sink placed infinitely close together and consequently satisfies the Laplace equation. These fundamental solutions will be discussed in section 1.4.
1.3 Vorticity and circulation

One of the simplifications of potential flow is its irrotational character. This means that fluid elements have no angular velocity and their motion is merely translation. For subsonic flows over wings the rotational character is often confined to the viscous boundary layer and can therefore be regarded as an irrotational problem.

The degree of rotation depends on the velocity field and is related via the vorticity vector $\mathbf{\xi}$ as shown in Equation 1.9. Since the vorticity is twice the angular velocity, a non-zero value of $\mathbf{\xi}$ represents a rotational flow field.

$$\mathbf{\xi} = \nabla \times \mathbf{V} \tag{1.9}$$

Related to this quantity is circulation, which is fundamental to calculate lift since Kutta-Joukowski theorem dictates that lift is a function of circulation $\Gamma$, shown by Equation 1.10.[10]

$$L' = \rho_\infty V_\infty \Gamma \tag{1.10}$$

Circulation $\Gamma$ is defined as the line integral around a closed curve of a velocity field, as shown in Equation 1.11.

$$\Gamma = -\oint_c \mathbf{V} \cdot ds \tag{1.11}$$

With Stokes’ theorem this line integral can be converted to a surface integral as done in Equation 1.12. The circulation about a closed curve equals the vorticity integrated over the surface. This means if the flow is irrotational at all points on the enclosed surface, no lift can be generated.

$$\Gamma = -\oint_c \mathbf{V} \cdot ds = -\iint_S (\nabla \times \mathbf{V})dS = -\iint_S \mathbf{\xi}dS \tag{1.12}$$

Furthermore, the vorticity can be related to the potential formulation of Equation 1.2. Combining yields:

$$\mathbf{\xi} = \nabla \times \mathbf{V} = \nabla \times (\nabla \cdot \mathbf{\phi}) + \nabla \times (\nabla \times \mathbf{\psi}) = \nabla \times (\nabla \times \mathbf{\psi}) \tag{1.13}$$

This means no vorticity can be present with only a scalar representation of the vector field, hence no lift can be generated. Therefore the vector potential $\mathbf{\psi}$ must be used to represent the effect of vorticity on the flow. Through vector calculus it can be shown that:

$$\mathbf{\xi} = \nabla \times (\nabla \times \mathbf{\psi}) \iff \mathbf{\xi} = \nabla^2 \mathbf{\psi} \tag{1.14}$$

Which is another Poisson equation, of which the solution can be obtained with Green’s function as outlined before.
1.4 Singularity elements

In previous sections some contradictions arose. In the Laplace equation for the scalar potential $\phi$, a forcing term was introduced which violates the mass conservation. However, sources and sinks in the flow are singular. They only add/remove flow at their center, which will be located on a body surface. These however will be used in such a way that the total mass flux of the system is 0.

Furthermore it was concluded that a vector potential $\psi$ was needed to generate vorticity, while potential flow is said to be irrotational. The vortex, being a solution to the Laplace equation, is also a singular. This means the rotational character is confined to the center of the vortex, while the rest of the flow field is irrotational. This is a physical representation of the flow field since vorticity is typically confined to the viscous boundary layer.

In this section these singularities will be described. The potential field of the fundamental solutions can be obtained through Green’s function and will be presented together with the according velocity field. Below the point source, point doublet and the point vortex are discussed. Since Laplace’s equation is a second-order differential equation, a linear function of position is obviously also a solution. An example of this type of solution is just a straight free stream flow.

1.4.1 Point source

A point source has streamlines running radially from one point as shown in Figure 1.1a. This means equipotential lines are concentric circles around the singularity point.

From the 3D formulation of Green’s function (Equation 1.8), the potential of a point source is defined as:

$$\Phi = -\frac{\sigma}{4\pi r}$$  (1.15)
1.4 Singularity elements

Where \( \sigma \) denotes the source strength and \( r \) the distance of the singularity point. Thus by the definition of the potential function, the velocity in spherical coordinates is:

\[
(q_r, q_\theta, q_\phi) = \left( \frac{\sigma}{4\pi r^2}, 0, 0 \right)
\]  

(1.16)

The potential of a sink singularity has an opposite sign, meaning that streamlines would flow towards the singularity point. The Cartesian form of Equation 1.15 is given below. This can be integrated over a line segment, surface or a volume to create other singularity elements, such as panels.

\[
\Phi(x, y, z) = -\frac{\sigma}{4\pi} \int_\mathcal{S} \frac{1}{\sqrt{(x-x_0)^2 + (y-y_0)^2 + (z-z_0)^2}}
\]

(1.17)

1.4.2 Point dipole

A point dipole consists of a source and a sink positioned infinitely close together. Due to their axial symmetry a primary unit vector \( \mathbf{e}_1 \) is defined, from sink to source. With \( \vartheta \) the angle between the unit vector \( \mathbf{e}_1 \) and the vector \( \mathbf{r} \), connecting the sink with the evaluation point, the potential is defined as:

\[
\Phi = -\frac{\mu \cos \vartheta}{4\pi r^2}
\]  

(1.18)

If the dipole axis is aligned with the \( x \) axis, \( \mathbf{e}_1 = \mathbf{e}_x \) and \( \vartheta = \theta \). So the potential in spherical coordinates becomes:

\[
\Phi(r, \theta, \phi) = -\frac{\mu \cos \theta}{4\pi r^2}
\]

(1.19)

When partially deriving Equation 1.19, the velocity components from Equation 1.20 are obtained.

\[
(q_r, q_\theta, q_\phi) = \left( \frac{\mu \cos \theta}{2\pi r^3}, \frac{\mu \sin \theta}{2\pi r^3}, 0 \right)
\]

(1.20)

The corresponding streamlines are shown in Figure 1.2. Since the quadrilateral source panels which are treated later are based on a Cartesian reference system, Equation 1.19 is given in the Cartesian reference system below:

\[
\Phi(x, y, z) = -\frac{\mu x}{4\pi ((x-x_0)^2 + (y-y_0)^2 + (z-z_0)^2)^{3/2}}
\]

(1.21)

This is when the dipole is directed in the \( x \) axis, for \( y \) or \( z \) direction the numerator should be replaced by \( (y-y_0) \) or \( (z-z_0) \) respectively.
10 Potential flow solutions

Figure 1.2: Streamlines for a dipole in $x$ direction

1.4.3 Point vortex

A vortical flow is present if the streamlines are in concentric circles about a given point and the velocity does not change along the streamline [11]. So the flow field around a vortex has a zero radial velocity and a nonzero tangential velocity which is inversely proportional the distance from the vortex $r$. Hence:

$$V_\theta = \frac{C}{r} \tag{1.22}$$

To determine the constant $C$ one can determine the circulation first. The line integral around a closed curve of a velocity field for the vortical flow reads:

$$\Gamma = - \oint_c V \cdot ds = -V_\theta 2\pi r \tag{1.23}$$

Combining Equation 1.22 and 1.23 yields Equation 1.24, which is the velocity distribution around the free potential vortex. Due to the influence of the circulation on the induced velocity, $\Gamma$ is called the strength of the vortex.

$$V_\theta = -\frac{\Gamma}{2\pi r} \tag{1.24}$$

Integrating along the tangential direction gives the velocity potential for a vortex element. The arbitrary constant $C$ can be put to zero.

$$\Phi = -\frac{\Gamma}{2\pi} \theta + C \tag{1.25}$$

The streamlines and the equipotential lines are shown in Figure 1.3a. The decay of the tangential velocity due to vortex is displayed in Figure 1.3b accordingly.
1.5 Kutta condition

The Kutta-Joukowski theorem from Equation 1.10 related the circulation to lift. Note that this circulation is an alternative way to think about lift generation. The physical origin of lift is still the pressure difference between the upper and lower side of the body. However, for a particular body there are an infinite amount of valid solutions depending on the choice of $\Gamma$, as depicted in Figure 1.4a and 1.4b. However, from experience it is known that nature fixes $\Gamma$ for an airfoil at a particular angle of attack.

From experimental flow visualisation [12] it was observed that when an airfoil is impulsively started, the flow tries to curl around the sharp trailing edge, as shown in Figure 1.4a. From inviscid theory it is known that the according velocity tends to infinity, which cannot be sustained in real flow. In the real case the stagnation point moves towards the trailing edge as shown in Figure 1.4b, such that the flow from the top and the bottom of the airfoil leaves smoothly at the trailing edge. In this steady state the velocity from the upper and lower side of the airfoil is equal for cusped trailing edges, or 0 for a finite angle trailing edge. In any case the difference between the two velocities is zero. And since the local jump in tangential velocity across the vortex sheet is equal to the local sheet strength. Therefore the vorticity strength $\gamma$ at the trailing edge can be written as:

$$\gamma(TE) = V1 - V2 = 0$$ (1.26)
Chapter 2

A 3D panel code

In this chapter a 3D panel code is discussed to solve aerodynamic steady state potential problems. This steady state solution does at one hand require an impermeability condition on the surface and on the other an accurate wake geometry to account for the downwash. For the latter an iterative wake alignment procedure will be used since time-stepped methods would require too much timesteps. Furthermore they can pose problems in terms of vorticity conservation. Note that this panel method will have the traditional quadratic scaling and a Fast variant of the code will be presented later in this thesis.

This chapter will first outline the method to arrive at a linear set of equations to solve for the body singularity strengths with a given wake geometry. Then an iterative approach to align the wake with the local flow field is suggested which eventually results in the final body singularity strengths. Various numerical schemes are assessed to convert these into surface velocities, which in turn are used to obtain the force coefficients. Finally more practical considerations of the code are given, which is useful for future users and developers.

2.1 Singularity elements

Panel methods belong to Boundary Element Methods (BEM), based on the distribution of singularities over the geometry. These can be used to impose a boundary condition such as impermeability (zero normal flow on the boundary surfaces) or the Kutta condition.

These singularities are fundamental solutions to the Laplace equation and due to the linearity of this equation, a superposition of solutions is also a solution. Therefore the solutions described before, such as the point source and dipole can be distributed on panels which are in turn a solution to the Laplace equation. The point vortex is usually distributed along a rectangular line, making a vortex ring.

Depending on the order of the panel method, singularities can be constant, linear or quadratic. Higher order methods require less panels for an accurate representation of
the flow field and can be advantageous to model impermeable surfaces. Due to the later implementation of a Fast Multipole Method (FMM), which can only deal with constant singularity strengths, linear or quadratic singularities are not considered.

2.1.1 Source panel

A quadrilateral source panel is a surface which is bounded by 4 straight lines on which sources with a constant strength $\sigma$ per area are distributed. If the corner point coordinates are $(x_1, y_1, 0) \ldots (x_4, y_4, 0)$, the potential at point $P(x, y, z)$ can be evaluated by integrating Equation 1.17, such that:

$$
\Phi(x, y, z) = -\frac{\sigma}{4\pi} \int_S \frac{dS}{\sqrt{(x-x_0)^2 + (y-y_0)^2 + z^2}}
$$

The integration and the partial derivations to obtain the velocity field can be found in Ref. [13]. The resulting flow field of a quadrilateral source panel is visualised in Figure 2.1. Note that coordinate transformations will be needed to allow for arbitrary $(x, y, z)$ corner points.

![Figure 2.1: The flow field around a quadrilateral source](image)

2.1.2 Dipole panels

The concept of a quadrilateral dipole panel is analogous to the quadrilateral source panel. Now dipoles which are pointing in the $z$ direction are distributed on the surface bound by $(x_1, y_1, 0) \ldots (x_4, y_4, 0)$. Then the potential at point $P(x, y, z)$ can be written as:

$$
\Phi(x, y, z) = -\frac{\mu}{4\pi} \int_S \frac{z dS}{((x-x_0)^2 + (y-y_0)^2 + z^2)^{\frac{3}{2}}}
$$
2.1 Singularity elements

Again the integration and according velocities can be found in Ref. [13]. The flow field due to these distributed dipoles on a panel is drawn in Figure 2.2. It can be seen that the distributed vorticity can be concentrated as discrete vortices at the bounding lines, the so-called vortex rings, as described next.

![Flow field around a quadrilateral dipole panel](image)

**Figure 2.2: The flow field around a quadrilateral dipole panel**

2.1.3 Vortex rings

Vortex rings are basically horseshoe vortices where the infinite length of the trailing vortices is given a finite value. As shown in Figure 2.3a, the vortex ring consists of 4 filaments. The forward segment (2-3) is called the bound vortex, the rear segment (1-4) the starting vortex and the ones linking the two are called trailing vortices (1-2 & 3-4).

![Vortex ring and nomenclature for induced velocity](image)

**Figure 2.3**

An expression should be found to estimate the flow field around such a vortex ring. It was shown before that vortices induce a velocity inversely proportional to distance \( r \).
The induced velocity field of an elemental vortex segment $dl$ with strength $\Gamma$ is given by Biot-Savart law \[13\], shown in Equation 2.3. Integrating this over a straight vortex filament yields Equation 2.4, the corresponding nomenclature can be seen in Figure 2.3b. Note that bold symbols represent vector quantities. Each vortex ring will thus have four contributions to the induced velocity at a certain point, which each can be calculated using Equation 2.4.

\[
dV = \frac{\Gamma}{4\pi} \frac{dl \times r}{|r|^3} \quad (2.3)
\]

\[
V_p = \frac{\Gamma}{4\pi} \frac{r_1 \times r_2}{|r_1 \times r_2|^2} r_0 \cdot \left( \frac{r_1}{r_1} - \frac{r_2}{r_2} \right) \quad (2.4)
\]

It was proven that a quadrilateral dipole panel is identical to the vortex ring when $\Gamma = \mu$ [14]

### 2.2 Linear set of equations

For each collocation point, a linear set of equations is set up to ensure impermeability taking into account all the singularities in the flow. Furthermore it was shown that the Kutta condition should be satisfied in order to obtain an unique solution.

#### 2.2.1 Impermeability condition

The geometry is panelled in flapwise and chordwise direction and the singularity panels are distributed on the panels. The collocation points, the location where the boundary conditions are specified, are taken as the mean of the corner points.

At the latter points a zero total normal velocity is desired such that the surface becomes a streamline of the flow field. This total normal velocity originates from the free stream, the dipole panels, the source panels and the wake. For now the wake is omitted for simplicity.

The normal velocities induced by the dipoles and sources can be calculated as a function of the unknown dipole strengths $\mu$ and the source strengths $\sigma$ respectively. So for each collocation point, the influence of all dipoles and sources is computed in terms of the unknown $\mu$ and $\sigma$. These so-called influence coefficients are denoted by $a$ and $b$ in Equation 2.5, where each row represents the boundary condition for one collocation point. The free stream is known and its normal component at each collocation point can be transferred to the right hand side (RHS) of the linear system of equations.

\[
\begin{bmatrix}
ad_{11} & a_{12} & \cdots & a_{1m} \\
ad_{21} & a_{22} & \cdots & a_{2m} \\
\vdots & \vdots & \ddots & \vdots \\
a_{m1} & a_{m2} & \cdots & a_{mm}
\end{bmatrix}
\begin{bmatrix}
\mu_1 \\
\mu_2 \\
\mu_3 \\
\vdots \\
\mu_m
\end{bmatrix}
+
\begin{bmatrix}
b_{11} & b_{12} & \cdots & b_{1m} \\
b_{21} & b_{22} & \cdots & b_{2m} \\
\vdots & \vdots & \ddots & \vdots \\
b_{m1} & b_{m2} & \cdots & b_{mm}
\end{bmatrix}
\begin{bmatrix}
\sigma_1 \\
\sigma_2 \\
\sigma_3 \\
\vdots \\
\sigma_m
\end{bmatrix}
=
\begin{bmatrix}
\text{RHS}_1 \\
\text{RHS}_2 \\
\vdots \\
\text{RHS}_m
\end{bmatrix} \quad (2.5)
\]
As mentioned, $a$ and $b$ denote the influence coefficients of dipoles and sources respectively. For example $a_{12}$ represents the influence of the second dipole panel ($\mu_2$) on the collocation point of the first panel. The right hand side (RHS$_1$) is the negative normal free stream velocity.

It can be seen in Equation 2.5 that for $m$ panels there are $2m$ unknowns, meaning that the system is not unique. Due to the linearity of the Laplace equation it is allowed to fix one of the unknowns. Therefore the source strengths are fixed as:

$$\sigma_k = n_k \cdot Q_\infty$$

(2.6)

In this way the source panels already counteract part of the normal velocity originating from the free stream. Therefore the dipole strengths can be be smaller and gradients will be lower, leading to higher accuracy. Since the source strengths are known, they can be transferred to the right hand side as illustrated in Equation 2.7.

$$
\begin{bmatrix}
a_{11} & a_{12} & \cdots & a_{1m} \\
a_{21} & a_{22} & \cdots & a_{2m} \\
a_{31} & a_{32} & \cdots & a_{3m} \\
\vdots & \vdots & \ddots & \vdots \\
a_{m1} & a_{m2} & \cdots & a_{mm}
\end{bmatrix}
\begin{bmatrix}
\mu_1 \\
\mu_2 \\
\mu_3 \\
\vdots \\
\mu_m
\end{bmatrix}
=
\begin{bmatrix}
\text{RHS}_1 \\
\text{RHS}_2 \\
\text{RHS}_3 \\
\vdots \\
\text{RHS}_m
\end{bmatrix}
$$

(2.7)

Note that the matrix containing the body-on-body dipole influences $a$ is often called the $A$ matrix. Such that:

$$A \cdot \mu = \text{RHS}$$

(2.8)

### 2.2.2 Kutta condition

Even though the system of Equation 2.7 is solvable, it is not physical since the Kutta condition is not embedded in these equations. As described before, this dictates that a body with a sharp trailing edge which is moving through a fluid will create about itself a circulation of sufficient strength to hold the rear stagnation point at the trailing edge [10]. Therefore the difference in circulation at the trailing edge between the upper and lower panel is shed in the wake.

Since each wake panel strength depends on the upper and lower trailing edge dipole strengths, the influence of the wake can be added to the $A$ matrix. In this way the linear system takes the Kutta condition explicitly into account. So additional to the $A$, which holds the body-on-body interactions, one can think about another matrix holding the influences of the wake on the body, called $A_{Kutta}$. With the strength being the difference between upper and lower trailing edge dipole strength, at the corresponding spanwise location.
Thus the influence of this wake panel row can be calculated on each panel as a function of the unknown vortex strengths of upper and lower trailing edge panels. In Equation 2.7 the influence coefficients $a$ will have an extra component if the panel is at the trailing edge to account for the contribution of the wake panel. Suppose for example that $a_k$ is a trailing edge panel, then the component $\sum_{i=1}^{m} c_{i,k}$ should be added. Here $c_{i,k}$ represents the influence of wake panel $k$ on panel $i$. The sign depends on whether the influenced panel is on top (+) or on the bottom (−) of the trailing edge.

### 2.3 Iterative wake method

Every lifting surface sheds a wake due to the generated circulation. To save computational effort, the wake shape is often prescribed. More realistic wake shapes can be obtained by aligning the wake with the local flow field. Since the wake cannot sustain a force, a wake configuration with zero normal velocity going through is sought. However, when changing the wake shape, the induced velocities at the collocation points of the body will be altered, changing in turn the linear set of equations to maintain the boundary condition at the body. Since this again affects the downstream flow field, the wake alignment should be done iteratively.

#### 2.3.1 Wake alignment

There are various options how to align the wake. The procedure below was found to work best in terms of stability and physical behaviour of the wake. Suppose two adjacent wake panels with common points $\vec{x}_1$ and $\vec{x}_2$ as depicted in Figure 2.4.

![Figure 2.4: Wake alignment nomenclature.](image-url)
A velocity is wanted to displace point $\vec{x}_2$ such that the common edge is aligned with the flow. This velocity is obtained by:

1. Calculate velocities $V_1$ and $V_2$ at panel centers
2. Set up axis systems $(t_1, m_1, n_1)$ and $(t_2, m_2, n_2)$ at panel centers
3. Interpolate velocities and axis systems to the center of the joining edge in $y$ direction. This yields a new velocity $V_c$ and axis system $(t_c, m_c, n_c)$.
4. Calculate $\vec{\delta} = \vec{x}_1 + \|\vec{x}_2 - \vec{x}_1\| \cdot \frac{\vec{V}_c}{\|\vec{V}_c\|} - \vec{x}_2$
5. Project $\vec{\delta}$ on the plane spanned by $\vec{m}_c$ and $\vec{n}_c$ to get $\vec{\delta}_d$
6. Displace $x_2$ and all downstream points of that spanwise instance with $\beta \vec{\delta}_d$, with $\beta$ being a preset relaxation factor.

At the edge of the wake, the velocity and axis system are taken the same as the ones at the last panel center point. Furthermore as can be seen from the procedure, the displacement is done marching downstream which was found to improve convergence considerably. Another practicality improving the wake stability is adding an extra long wake panel downstream of the last wake panels. Its displacement is based on the last panel located upstream at the same spanwise section. Finally the length of the first wake panel behind the trailing edge was doubled, this yields a more gradual roll-up and avoids singularities at early in the iteration process.

### 2.3.2 Convergence parameters

To check convergence a parameter should be defined which takes into account the residual force exerted on each panel. Keeping in mind that the latter is the mass flux times the velocity, the wake residual is defined according to Equation 5.1. Note that $S_i$ denotes the interpolated surface in $y$ direction.

$$ r_w = \sqrt{\frac{\left( \sum_{i=1}^{N} (S_i \cdot V_{n_i}^2) \right)^2}{\sum_{i=1}^{N} (S_i)}} $$  \hspace{1cm} (2.9)  

Furthermore it is convenient for convergence studies to have a residual based on the displacement $\vec{\delta}_d$ as well. So the solver residual is defined as:

$$ r_s = \sqrt{\frac{\left( \sum_{i=1}^{N} (S_i \cdot \|\vec{\delta}_d\|) \right)^2}{\sum_{i=1}^{N} (S_i)}} $$  \hspace{1cm} (2.10)
2.4 Secondary calculations

2.4.1 On-surface velocities

With the known circulations the loads can be calculated. First the induced velocities in the control points should be determined taking into account the influence of the body, wake and external conditions. Note that the tangential velocity is smooth only some distance from the surface, one panel length or more. This means that the tangential velocity is underestimated in the collocation point, as shown in Figure 2.5.

![Figure 2.5: Surface velocity for discrete vorticity panels](image)

At the control points there is almost no tangential velocity due to the nearby vortices. If one wants to calculate the tangential velocity at the control point of the continuous vorticity sheet of Figure 2.6, one should also take the discontinuous jump across the surface into account.

![Figure 2.6: Surface velocity for continuous vorticity sheet](image)

This jump is equal to half of the local strength of vorticity and is called the principal value, which must be added when calculating on surface tangential velocities. For discrete vorticity panels this principal value equals \(-0.5\nabla \Gamma\), based on the vorticity strength at neighbouring panels. Using the lay-out of Figure 2.7, the gradient of the vorticity strength can be determined using Equation 2.11.

\[
\nabla \Gamma \approx \frac{\left[ (r_w - r_n)(\Gamma_w + \Gamma_n) + (r_s - r_w)(\Gamma_s + \Gamma_w) \right] \times n_c}{\left\| (r_n - r_s) \times (r_w - r_e) \right\|} \nabla \Gamma
\]

(2.11)

2.4.2 Force coefficients

Then the pressure coefficient \(C_p\), defined by Equation 2.12 is determined. For incompressible, inviscid and steady flow Bernoulli’s equation can be used and the pressure coefficient
can be expressed as Equation 2.13

\[ C_p = \frac{p - p_\infty}{0.5\rho_\infty V_\infty} \]  

\[ C_p = 1 - \left( \frac{V}{V_\infty} \right)^2 \]  

From pressure coefficient the force coefficients can be calculated easily. The nondimensional local fluid dynamic load in normal direction for panel \( k \) now equals [13]:

\[ C_{F_k} = - \frac{C_{p_k} S_k}{S} \cdot n_k \]  

Here \( S_k \) is the panel area and \( S \) is a reference area, mostly taken as the projected area of the lifting surface. From this load in normal direction, other components can be calculated to isolate lift and drag.

\section*{2.5 Implementation}

The method described above or similar to it has been used successfully in commercial panel codes such as CMARC and VSAero. These codes are not free to the public and source code modifications are tedious. Therefore a structured, open-source and free to use implementation of this panel code can serve the scientific community. A new panel code implementation will be done in a MATLAB environment, keeping in mind user friendliness.

A vortex lattice MATLAB implementation for linear aerodynamic wing applications was developed by Melin [15], which is the well-known \textit{TORNADO} code. Mata Bueso [16] made an unsteady version of this code by timestepping the wake and called this version...
Both codes can deal with arbitrary geometries which are modelled as mean camber lines. Even though these vortex lattice codes cannot model thickness, the steps taken to arrive at a solution are similar. Therefore these codes were used to get acquainted with the solving methods, which is reflected in the final structure of the developed code.

In Figure 2.8 the main blocks of the panel code are shown. In the presolver, the inputs from the user are used to generate a lattice of singularity panels. The geometry is defined using the routine \texttt{fgeo} while the external conditions and solver properties are specified in \texttt{fstate}. The geometry and the state are used in \texttt{flattice} to generate the lattice. In \texttt{fsolver} the linear system of equations is set up and solved, then the wake is aligned with the local flow iteratively. Finally the postprocessor calculates the pressure distribution and the loads on the wing, which can be shown in a graphical output.

### 2.5.1 Presolver

The first step in the presolver is the function \texttt{fgeo} where the geometry is defined. Currently these geometry types can be defined: a 3D wing, a plane, a cylinder and an extruded square. The first requires the position, dimensions, angle of attack, airfoil type, discretization etc. while the others just require position, dimensions and discretization. This discretization is defined by specifying the number of panels in different directions. One can choose a uniform, cosine or double cosine distribution.

Following information about the external conditions such as free stream velocity and density is provided in \texttt{fstate}. Also solver properties such as the number of wake iterations,
2.5 Implementation

number of wake steps and relaxation factors are defined here.

Then the above information is used to make a lattice in flattice. The first step is to
discretize the airfoil coordinates which was done based on the path length. The result
can be seen in Figure 2.9 for a Clark Y airfoil.

![Figure 2.9: Airfoil discretization](image)

The airfoil coordinates are then scaled using the corresponding chord and rotated to the
right incidence angle. Next these are distributed in spanwise direction according to the
selected spanwise mesh, which yields the 3D geometry of Figure 2.10, showing 6 chordwise
and 12 spanwise panels. In flattice also the normals are calculated, pointing out of the
airfoil.

![Figure 2.10: 3D geometry with panel index numbers](image)

Note that the panel data for both wings is stored in 1D vectors and mapping functions
are used to rearrange when needed. For the 2D airfoil, counting starts at the bottom
trailing edge in clockwise direction to the top trailing edge. Counting is done per section
and starts from the center towards the tip. Then for the mirrored part the same logic
holds, from the root to the tip, one spanwise cross-section at the time. In Figure 2.10 one
can see some index numbers for this geometry to illustrate the index mapping.
2.5.2 Solver

The solver is the most elaborate module of the code and the main components are visualised in Figure 2.11. First a wake shape is assumed, being a planar vorticity sheet aligned with the free stream flow. Then in fcoeff the velocities at each collocation point are calculated using the subroutines SourceQ and VortxR as suggested by Katz and Plotkin [13]. Since these influences can be calculated independently, this is done in a parallel loop. The body-on-body influences and the wake-on-body influences in function of the unknown body vorticity strengths are stored in matrix $A_{\text{basic}}$ and $A_{\text{Kutta}}$ respectively. The known normal velocities at each panel due to the free stream and the sources are saved in $\text{RHS}$.

The sum of $A_{\text{basic}}$ and $A_{\text{Kutta}}$ then determines the linear set of equations. A sanity check of these coefficients can be done by making a surface plot of the induced velocity magnitudes, as shown in Figure 2.12 and 2.13 for a wing with 5 chordwise panels at each side and 20 spanwise panels. Color patterns appear according to the indexing method which is used. For the surface plot of the body influences for example, the diagonal represents the self influence of each panel, hence the bright color. Since the Kutta condition is merely a function of the strengths of the trailing edge panels a few columns of this matrix are filled, shown as the bright color bands in Figure 2.13.

With these matrices and the right hand side, the vorticity strengths of the body can be calculated which are needed in the wake displacement function $f_{\text{wakedisp}}$. The function $f_{\text{Vind}}$ is used whenever one wants to know the total induced velocity by all the sources and dipole panels, so also here to calculate the velocity at the center of each wake panel. Then the updated wake geometry is fed into the solver again to calculate the updated $A_{\text{Kutta}}$, note that $A_{\text{basic}}$ and the $\text{RHS}$ did not change so do not require new evaluation. This iterative process is done until the maximum amount of iterations, specified in state, is reached.

2.5.3 Post-processor

In fpostpro, the results for each iteration can be analysed and visualised. These are a few tools which are helpful for the post processing.

- $f_{\text{plotwake}}$ plots the geometry and the wake at a specified iteration count.
- $f_{\text{planevel}}$ calculates the in- and out of plane velocities at a plane specified by the corner points and the resolution of points which can be used to make a surface plot.
- $f_{\text{Xfoil cp}}$ loads the geometry at a spanwise location in XFOIL [17] leading to a reference pressure distribution.
Figure 2.11: Solver flow chart
Figure 2.12: Body-on-body influence coefficients

Figure 2.13: Wake-on-body influence coefficients
During the development of the basic 3D panel code it became clear that numerical times were impractically high for high panel counts. Three main contributors can be distinguished. The first is the calculation of the \textit{body-on-body influences} the first iteration, the second are the \textit{wake-on-body influences} each iteration and finally the calculation of the wake velocities to align the wake with the local velocity. Each calculation scales with a higher order of $N$, leading to these high calculation times. In this chapter some methods are suggested which can be used to reduce the order with which these calculations scale. These methods are often referred to as Fast algorithms.

First the theory of a Fast Multipole Method is explained, followed by a demonstration of an existing FMM library. Then iterative solvers are considered and some convergence characteristics are looked at. The FMM does not accelerate the calculation of the $A$ matrix since the isolated influence of each panel on each collocation point is needed. Therefore a FastBEM procedure is discussed, which couples the FMM with these iterative solvers and bypasses the requirement to set up a linear set of equations.

\section{3.1 Fast Multipole Method}

If $N$ singularity panels are used, $N^2$ operations are needed to calculate the induced velocity on each panel due to the other panels. This means that computational times will increase significantly for large number of panels. Greengard and Rokhlin \cite{5} developed an algorithm to rapidly evaluate potential and force fields which involve a large number of particles which interact with each other. Using this so called Fast Multipole Method (FMM) the asymptotic CPU time estimate is of the order $O(N)$. Therefore it became a well established tool to numerically simulate potential fields in plasma physics, molecular dynamics, fluid dynamics and celestial mechanics.

The algorithm makes use of the multipole expansion, a mathematical series which can represent a function depending on angles or radii. The series can be truncated such that a certain precision is obtained. This will be shown in subsection 3.1.1.
The method relies on the subdivision of the 3D potential field in cubical boxes, and distributed particles within these boxes are clustered. The 2D routine will be treated to illustrate the idea, in this case particles are clustered in squares instead of cuboids. Then some results are given in subsection 3.1.3, which were obtained by testing an existing FMM library.

3.1.1 Theory

The multipole expansion formulates the potential $\phi$ in a field of charges, or dipoles and sources in this case, with strengths $\{q_i, i = 2, ..., m\}$ located at points $\{z_i, i = 2, ..., m\}$ with $\|z_i\| < r$. Then for any point $x + iy = z \in \mathbb{C}$ with $\|z\| > r$, the potential $\phi(z)$ is given by equation Equation 3.1 [5].

$$\phi(z) = Q \log(z) + \sum_{k=1}^{\infty} \frac{a_k}{z^k} \quad (3.1)$$

where

$$Q = \sum_{i=1}^{m} q_i \quad \text{and} \quad a_k = \sum_{i=1}^{m} \frac{-q_i z^k}{k} \quad (3.2)$$

Also, for any $p \geq 1$

$$\left| \phi(z) - Q \log(z) - \sum_{k=1}^{p} \frac{a_k}{z^k} \right| \leq \alpha \left| \frac{r}{z} \right|^{p+1} \leq \left( \frac{A}{c - 1} \right) \left( \frac{1}{c} \right)^p \quad (3.3)$$

where

$$c = \left| \frac{z}{r} \right|, \quad A = \sum_{i=1}^{m} |q_i| \quad \text{and} \quad \alpha = \frac{A}{1 - |r/z|} \quad (3.4)$$

This means that the truncation error can be controlled and predicted using variable $p$. If $c \geq 2$, the error bound can be obtained through:

$$\left| \phi(z) - Q \log(z) - \sum_{k=1}^{p} \frac{a_k}{z^k} \right| \leq A \left( \frac{1}{2} \right)^p \quad (3.5)$$

In the next 2D example it will be shown how the multipole expansion will reduce the computational effort. Imagine a set of charges with strengths $q_1, q_2, ..., q_m$ at points $x_1, x_2, ..., x_m \in \mathbb{C}$ and a set of points $y_1, y_2, ..., y_n \in \mathbb{C}$. These two sets of points are well separated if they comply with Figure 3.1 or algebraically, if there exist points $x_0, y_0 \in \mathbb{C}$ and a real $R > 0$ for which:

$$|x_i - x_0| < R \quad \text{for all} \quad i = 1, ..., m$$
$$|y_j - y_0| < R \quad \text{for all} \quad j = 1, ..., n$$
$$|x_0 - y_0| > 3R \quad (3.6)$$
3.1 Fast Multipole Method

To know the potential at the set of points \( \{y_j\} \) due to the charges at \( \{x_i\} \) one would apply Equation 3.7.

\[
\sum_{i=1}^{m} \phi_{x_i}(y_j) \quad \text{for all } \quad j = 1, ..., n \tag{3.7}
\]

This requires evaluating \( m \) fields at \( n \) points, thus order \( mn \) work. The multipole expansion of the potential due to the charges \( q_1, q_2, ..., q_m \) about \( x_0 \) is shown in Equation 3.8.

\[
\left| \sum_{i=1}^{m} \phi_{x_i}(y_j) - Q \log(|y_j - x_0|) - \sum_{k=1}^{p} \frac{a_k}{|y_j - x_0|^k} \right| \leq A \left( \frac{1}{2} \right)^p \tag{3.8}
\]

The \( p \)-term multipole expansion of the potential due to \( q_1...q_m \) requires work \( O(mp) \) and evaluating this expansion at all points \( y_1...y_n \) needs a number of operations scaling with \( O(np) \). In order to have a relative precision \( \epsilon \) w.r.t. the total charge, \( p \) should be of order \( -\log_2(\epsilon) \). Then the order of the computation is \( O(m + n) \), which reduces the original computation time where the order was \( O(mn) \). This means if we double the points in set \( \{x_i\} \) and \( \{y_j\} \), the computational time is doubled instead of squared which will speed up the calculations for a large number of points. The numerical implementation was done for the 2D and 3D case in [5] and [18] respectively.

3.1.2 Method

In the FMM, the 3D potential field is subdivided in cubical boxes, and then distributed particles within these boxes are clustered. To illustrate the idea, the 2D routine will be treated in which the particles are clustered in squares instead of cuboids.

The extent of refinement is defined by the level \( l \), a level of 0 is the entire domain while a level 1 subdivides this domain box in 4 equal parts. Each box is again divided in 4 boxes for a higher level, which means the total number of boxes is \( 4^l \).

In the simple example it was shown that particles should be well separated. Which means the multipole method cannot be applied for neighbouring boxes. As can be seen in Figure 3.2a, the multipole expansion can be used to compute the interactions between...
particles in box X and the white boxes. Then a recursive approach is used to determine the interactions between particles in each box and the ones in the near neighbours.

In Figure 3.2b it can be seen that the next step is to increase the refinement by one level. This means the interaction between the box marked with X and the well separated white boxes can be computed using the multipole expansion. Note that the interaction with the light grey boxes was already covered in the previous step. This is done for roughly $\log_4(N)$ levels of refinement which results to $4^{\log_4(N)} = N$ number of boxes, or $O(1)$ particles by box when assuming an uniform distribution.

3.1.3 Results

Since a panel code is based on potential flow theory the FMM can be used to speed up the process. Greengard and Gimbutas [19] implemented the fast multipole method to deal with problems governed by the Laplace or Helmholtz equation. The provided FMM library can deal with particles and triangles as singularity geometries. Furthermore sources and dipoles could be used as singularity type.

To check the efficiency of the method, the algorithm was tested for various panel counts and precision flags. Since each quadrilateral will be represented by two triangles, only the case with triangular singularities was considered to see when the use of an FMM becomes beneficial.

In Figure 3.3 the calculation times are shown for a range of number of triangles ($N$) and precision flags. Each precision flag corresponds to an accuracy as shown in Table 3.1. It can be seen that for very low panel numbers, the numerical time is dominated by the generic calculations, which means it is rather independent of $N$. From approximately 100 panels, a more linear (logarithmic) behaviour is observed. For these panel numbers the FMM is always slower than the direct method. From approximately 2200 panels onwards, the FMM has a time advantage for all accuracies. As expected, the numerical time increases for higher accuracy.

In Table 3.1 the efficiency for different precision flags ($iprec$) is summarized. In the second column the according accuracies are shown, a precision flag of 0 corresponds for
3.1 Fast Multipole Method

![Figure 3.3: Calculation times: FMM vs. direct method](image)

<table>
<thead>
<tr>
<th>Iprec</th>
<th>Accuracy</th>
<th>Logarithmic slope</th>
<th>Break even point</th>
</tr>
</thead>
<tbody>
<tr>
<td>-2</td>
<td>(\pm 0.5 \cdot 10^0)</td>
<td>0.92</td>
<td>327</td>
</tr>
<tr>
<td>-1</td>
<td>(\pm 0.5 \cdot 10^{-1})</td>
<td>0.90</td>
<td>247</td>
</tr>
<tr>
<td>0</td>
<td>(\pm 0.5 \cdot 10^{-2})</td>
<td>0.96</td>
<td>250</td>
</tr>
<tr>
<td>1</td>
<td>(\pm 0.5 \cdot 10^{-3})</td>
<td>1.04</td>
<td>325</td>
</tr>
<tr>
<td>2</td>
<td>(\pm 0.5 \cdot 10^{-6})</td>
<td>1.15</td>
<td>800</td>
</tr>
<tr>
<td>3</td>
<td>(\pm 0.5 \cdot 10^{-9})</td>
<td>1.23</td>
<td>1220</td>
</tr>
<tr>
<td>4</td>
<td>(\pm 0.5 \cdot 10^{-12})</td>
<td>1.28</td>
<td>1370</td>
</tr>
<tr>
<td>5</td>
<td>(\pm 0.5 \cdot 10^{-15})</td>
<td>1.31</td>
<td>1708</td>
</tr>
<tr>
<td>Direct</td>
<td>n.a.</td>
<td>1.85</td>
<td>n.a.</td>
</tr>
</tbody>
</table>

The next column contains the logarithmic slopes of the trendlines of the quasi-linear part, which generally goes up with increasing accuracy. The direct method has the largest slope, meaning that the numerical time increases with \(O(N^{1.85})\).

The next column contains the break even point, the point where using FMM is more time-efficient than the direct method. Note that these are triangular panels, so for thus for quadrilateral source and dipole panels this number should be halved. So using FMM in a panel code is useful from 685 panels onwards for an accuracy of 12 digits.

In the previous table, mutual interactions were investigated. But in case of velocity field evaluation the number of targets can be different from the amount of singularity panels. Therefore the number of targets was varied for different amounts of singularity panels. As can be seen in Figure 3.4, both the number of targets and the number of singularity panels scale the evaluation time in a constant logarithmic manner when both are significantly high (\(N_{\text{Targets}}\) and \(N_{\text{Panels}} > 200\)).
To estimate the computational cost one can set up an equation for the logarithmic plane of Figure 3.4, as shown below for iprec 3 and 5 respectively.

**iprec 3**

\[
\log(T) = 0.61 \log(N_{\text{Panels}}) + 0.51 \log(N_{\text{Targets}}) - 3.75 \\
\iff T = 1.78 \cdot 10^{-4} \cdot N_{\text{Panels}}^{0.61} \cdot N_{\text{Targets}}^{0.51}
\]  

(3.9)

**iprec 5**

\[
\log(T) = 0.69 \log(N_{\text{Panels}}) + 0.63 \log(N_{\text{Targets}}) - 4.31 \\
\iff T = 4.90 \cdot 10^{-5} \cdot N_{\text{Panels}}^{0.69} \cdot N_{\text{Targets}}^{0.63}
\]  

(3.10)

If the number of panels and targets would be the same, one can add the powers of \(N_{\text{Panels}}\) and \(N_{\text{Targets}}\). For iprec 5 this is 1.32, which is close to the corresponding logarithmic slope found in Table 3.1.

It is expected that the time gain for the panel code will be even larger. In this particular test triangles were distributed randomly within a cubical constraint. This was done a 10 times and the times were averaged since numerical times will vary depending on the local density of triangles. In a panel code where a wake evolves from a wing, a more elongated boundary is created. This allows for more clustering leading to a greater time advantage. Therefore the use of FMM will significantly improve the time efficiency of a panel code.

### 3.2 Iterative solvers

Inverting large asymmetric matrices to solve a linear system of equations is numerically expensive. Often iterative solving methods are used to deal with these. Furthermore, they can be coupled with the FMM algorithm to efficiently solve for the unknowns, as
will be shown in section 3.3. Therefore their theory is discussed below, together with some convergence properties and computational experiments.

Iterative solvers improve approximated solutions until a preset criterion is met, this is usually the maximum number of iterations or a tolerance w.r.t. the exact solution. One can distinct stationary and non-stationary methods, of which the former are usually the older, simpler and least effective ones. Examples are the Jacobi method, Gauss-Siedel or Successive Over-relaxation (SOR) method. Non-stationary methods are more complex and mostly make use of orthogonal bases.

The rate at which methods converge largely depends on linear system one is trying to solve. Unlike the direct solver, CPU time for iterative solvers are hard to predict. Some cases can converge in a few iteration while others need hundreds, or fail to converge completely. It was found that the number of iterations depend greatly on the condition number of the matrix, which can partly be controlled by preconditioning the system. This transforms the coefficient matrix in a more favourable one without adding much computational time.

### 3.2.1 Preconditioning

As mentioned in before, a good preconditioner improves the convergence of an iterative solver without adding much computational effort. Without preconditioning, convergence to the required tolerance will take too long or might even be impossible to obtain. Preconditioners are used to change the condition number of the matrix, which denotes how much the output changes with varied input. Thus if the condition number $\kappa$ is large, a small error in the right hand side might create a large error in the unknown. A very large $\kappa$ denotes an ill conditioned system, where influence matrix $A$ is close to being singular. The formulation of condition number $\kappa$ of influence matrix $A$ can be found in Equation 3.11.

$$\kappa(A) = \|A^{-1}\| \cdot \|A\|$$  \hspace{1cm} (3.11)

Preconditioning can reduce this condition number. So if one wants to solve the linear system $Ax = b$, one can choose a nonsingular premultiplier matrix $P$ such that:

$$AP^{-1}P^{-1} = b$$  \hspace{1cm} (3.12)

With $y = P^{-1}x$ this can be simplified to:

$$AP^{-1}y = b$$  \hspace{1cm} (3.13)

Equation 3.14 represents the right preconditioned system. In the left preconditioned system terms are moved to the left side, yielding:

$$P^{-1}(Ax - b) = 0$$  \hspace{1cm} (3.14)
Fast algorithms

The cheapest preconditioning matrix is the unitary matrix, but does not affect the condition number. And using the $A$ matrix as preconditioner will yield the ideal condition number of 1, meaning it converges in one iteration. However this is equivalent to the direct method and will be numerically expensive. Finding the ideal preconditioner can be a tedious task. Often the diagonal of the $A$ matrix is used. Also an incomplete LU factorisation can be done such that $A \approx LU$, and the matrix $P$ can explicitly be set to this $LU$ product.

3.2.2 Jacobi

Below Jacobi’s iterative method is shown, due to its simplicity it is a good introduction to the concept of iterative solvers. For a linear system of equations $Ax = b$, the $k$-th iterative step is defined according to Equation 3.25.

$$x^{k+1} = D^{-1}(b - Rx^k)$$  \hspace{1cm} (3.15)

Here $D$ is the diagonal matrix of $A$, and $R$ is the $A$ matrix with the diagonal set to zero, such that $A = R + D$. For the first iteration, an initial guess for $x$ has to be set, which is often set to the zero solution. One should note that this method is only guaranteed to converge if matrix $A$ is diagonally dominant, meaning that the absolute value of the diagonal should be larger than the sum of all other values on the same row. The influence matrix of a panel method does not have this property since the induced velocity by the sum of all other panels might be larger than the self influence.

3.2.3 GMRES

Since the Jacobi method is not guaranteed to converge for the considered problems, one should look at other solution methods. The Generalised Minimised Residual (GMRES) method is applicable on any nonsymmetric system of linear equations. This is a nonstationary iterative method, meaning that its computations involve information that changes each iteration. Constants can for example be calculated using the inner product of residuals. Below the main structure of the algorithm is shown, for detailed information one should consider Ref. [20].

In the Jacobi method it can be seen that the final iterative solution is a linear combination of $A^{k-1}b$, with $k$ is ranging from 1 to the maximum number of iterations. This span is defined as a Krylov subspace, the $n$-th Krylov can thus be written as:

$$K_n(A, b) = \text{span} \{ b, Ab, A^2b, \cdots, A^{n-1}b \}$$ \hspace{1cm} (3.16)

This basis is not linearly independent due to the way they are defined. Therefore it is useful to convert this basis to a set of orthonormal vectors $Q$. The vectors $q_1, q_2, \ldots, q_n$ can be formed explicitly using the Arnoldi method [21], of which the pseudocode is shown below:
% Arnoldi iteration
q_1 = b/\|b\|
for n = 1, 2, 3, ...
    v = Aq_n
    for j=1:n
        h_{jn} = q_j \cdot v
        v = v - h_{jn} q_j
    end
    h_{n+1,n} = \|v\|_2
    q_{n+1} = v / h_{n+1,n}
end

The most expensive step in the algorithm is the matrix-vector multiplication \( Aq_n \), which is in the order of \( O(Nn) \), with \( N \) the number of unknowns and \( n \) the number of orthonormal basis vectors. Furthermore it can be seen that the algorithm does not need \( A \) explicitly, meaning that the product \( Aq_n \) can be used as a black box, such as the FMM. This is known as the FastBEM method, discussed in section 3.3:

Since the solution \( x_n \in \mathcal{K}_n \), it can be written as a linear combination of basis vectors \( q_1, ..., q_n \). If matrix \( Q \) has those vectors as columns, then \( x_n = Q_n y_n \) with \( y_n \in \mathbb{R} \). Now the goal is to find \( y_n \) such that the residual norm is minimized:

\[
\|Ax_n - b\| = \|AQ_n y_n - b\| \rightarrow \text{min}
\] (3.17)

This expression can be simplified. Next to the matrix \( Q_n \), also a matrix \( \tilde{H}_n \) is an output of the Arnoldi iteration. This is an \((n + 1)\) by \( n \) upper Hessenberg matrix, meaning that entries are zero below the first subdiagonal. This matrix is related to \( A \) by:

\[
AQ_n = Q_{n+1} \tilde{H}_n
\] (3.18)

This means Equation 3.17 can be written as:

\[
\|Q_{n+1} \tilde{H}_n y_n - b\| \rightarrow \text{min}
\] (3.19)

Since the norm does not change due to the multiplication with an orthogonal matrix, one can multiply the left term with \( Q_{n+1}^{-1} \), yielding:

\[
\|\tilde{H}_n y_n - Q_{n+1}^{-1} b\| \rightarrow \text{min}
\] (3.20)

Then the product \( Q_{n+1}^{-1} b \) can be rewritten, starting from Equation 3.21:

\[
Q_{n+1}^{-1} b = \begin{bmatrix} q_1^{-1} b \\ q_2^{-1} b \\ \vdots \\ q_{n+1}^{-1} b \end{bmatrix}
\] (3.21)
Since the the columns $q_j$ of matrix $Q_n$ are an orthonormal basis for Krylov space $\mathcal{K}_n$, can $q_1$ be written as $q_1 = \frac{b}{\|b\|}$. Due to the orthonormal properties, $q_{j+1} = b = 0$ for any $j > 1$. Which means the product $Q_n^{-1}b = 0$ can be written as $\|b\|e_1$. This leads to the final least squares formulation of Equation 3.22. This least squares problem can for example be solved using a QR factorisation [22]

$$\|\tilde{H}_ny_n - \|b\|e_1\| \rightarrow \text{min.} \quad (3.22)$$

Now the main structure of the GMRES can be destilled:

```matlab
% GMRES
q_1 = b/\|b\|
for n = 1, 2, 3, ...
    Perform \(n\)-th step of Arnoldi iteration and calculate \(\tilde{H}_n\) and \(Q_n\)
    Find \(y\) that minimizes \(\|\tilde{H}_ny_n - \|b\|e_1\|
    Check residual norm
    \(x_n = Q_ny\).
end
```

The major drawback of GMRES is that the amount of work and storage for each iteration scales linearly with the iteration count. So unless really fast convergence is reached, this cost may become prohibitive [23]. This can be solved by restarting the iteration, after a preset $m$ number of iterations, the data is cleared and set as initial data for the next $m$ iterations. The numerical efficiency of the GMRES was proven to depend primarily on the value of $m$, as shown by [20]. If $m$ is too small, GMRES might be slow to converge or fail to converge entirely. However, if it is larger than the required $m$, work and memory requirements increase. Determining $m$ is a matter of experience since there are no definite rules to predict it.

### 3.2.4 Results

Now the GMRES algorithm as suggested by [23] will be tested for various cases. Of particular interest is the effect of preconditioning and the condition number on the convergence properties. In Figure 3.5 the residual norm for each iteration is shown, with a tolerance of $10^{-10}$. The input matrices are random matrices conditioned to a preset condition number $\kappa$. Note that the condition number of the influence matrix of the panel code typically range between $10^4$ to $10^7$. As expected, one can see improved convergence for low condition numbers.

Furthermore, the GMRES stagnates for a condition number of $10^4$ and higher. In Figure 3.5b the diagonal of $A$ is set as the preconditioner, which improved the solution, however still no convergence for condition number $10^4$ is reached. Then the preconditioner is expanded to the diagonal together with two sub- and two superdiagonals of the
A, which improved the convergence of the case with $\kappa = 10^4$. Finally the most expensive preconditioner, namely the LU decomposition, is applied in Figure 3.5d. This yields excellent convergence to the solution in a small amount of iterations.

One can see that the highest original condition number does not correspond to the worst convergence properties anymore. This is because the condition number does not scale consistently when premultiplying with a LU factorisation. Even though it is numerically the most expensive method per iteration, the excellent convergence properties make it a time efficient method, especially for large condition numbers.

![graph](image)

**Figure 3.5:** Convergence of GMRES algorithm to solve $A_{\text{basic}} \cdot x = b$ for different condition numbers $\kappa$ and preconditioners

From this analysis it can be concluded that choosing the preconditioner is of critical importance to reach convergence, or to reach convergence in an efficient way. Furthermore it was found that the number of unknowns merely affected the numerical time and not the convergence properties.

The ultimate application for GMRES in panel methods is to not explicitly set up $A$ during the iterative wake procedure as will be explained in section 3.3. To that end it is useful to know if matrix $A_{\text{basic}}$ is a good conditioner to solve the matrix $(A_{\text{basic}} + A_{\text{kutta}})$. An arbitrary kutta matrix is created and added to $A_{\text{basic}}$ with the previous condition num-

---

**3.2 Iterative solvers**

---
bers. This increases the condition number of the total system and decreasing convergence properties accordingly, as shown in Figure 3.6a. Furthermore it is not guaranteed that the best converging $A_{\text{basic}}$ will have the best convergence properties with $(A_{\text{basic}} + A_{\text{kutta}})$. This is proven by the case where $\kappa = 1$, and tells that convergence is unpredictable when applying matrix operations.

Therefore it is attempted to precondition the complete system with $A_{\text{basic}}$. It can be concluded that the preconditioners based on $A_{\text{basic}}$ are suitable for the complete $(A_{\text{basis}} + A_{\text{kutta}})$ problem. Again the $LU$ factorisation is the most effective due to the low iteration count. Note that the GMRES algorithm is still slower than the direct inverse of influence matrix. Only for very large panel numbers this might be beneficial. However the GMRES was not presented with the intent to solve an ordinary linear system of equations. A useful application for GMRES in panel methods will be discussed in section 3.3.

Figure 3.6: Convergence of GMRES algorithm to solve $(A_{\text{basic}} \cdot x = b)$ for different condition numbers $\kappa(A_{\text{basic}})$ and preconditioners.
3.3 FastBEM

In previous section it was shown how the FMM algorithm can be used to speed up velocity calculations by clustering points. However, in the influence matrix $A$ the contribution of each isolated singularity is needed, which still requires $O(N^2)$ operations, with $N$ the number of unknowns. Therefore one might consider a FastBEM approximation, which is can accelerate the Boundary Element Method (BEM) such as a panel code. Here the influence matrix is not defined explicitly, but the solution is iteratively found using the FMM algorithm. To demonstrate the concept, the FMM algorithm will be coupled to a Jacobi iterative solver. Then a more elaborate iterative solver, namely the generalized minimum residual method (GMRES), will be investigated.

3.3.1 FMM and Jacobi

The Jacobi iterative method can solve a diagonally dominant linear set of equations. The linear system of equations is defined as:

$$Ax = b$$  \hspace{1cm} (3.23)

For the Jacobi method, the $k$-th iterative step is defined according to Equation 3.25.

$$x^{k+1} = D^{-1}(b - Rx^k)$$  \hspace{1cm} (3.24)

Here $D$ is the diagonal matrix of $A$, and $R$ is the remainder such that $A = D + R$. This means that Equation 3.25 can be rewritten such that:

$$x^{k+1} = D^{-1}(b - Ax^k + Dx^k)$$  \hspace{1cm} (3.25)

This requires to calculate the diagonal of the a matrix directly, which is the influence of a singularity panel on itself, thus $N$ operations. The product $Ax$ can be calculated using the FMM, which is theoretically also proportional to $O(N)$, as shown before. This procedure has to be iterated until convergence is reached, so the whole process to iteratively determine $x$ is proportional to $O(N)$.

It was established that these stationary iterative methods, where the coefficients for the next step are independent of $k$, are not suitable for these problems since they require a specific type of $A$ matrix to ensure convergence. Therefore the GMRES algorithm was introduced, which can also be coupled with an FMM.
3.3.2 FMM and GMRES

It was found that the GMRES is a suitable iterative solver for the linear set of equations of a panel method. Furthermore it has the property, like the Jacobi method, that \( Ax \) can be used as a black box. This means that matrix \( A \) does not need to be defined explicitly and the product \( Ax \) can be calculated by other means, such as the FMM.

In this section the direct method will be compared to FMM coupled iterative method. To do so, singularity particles are positioned on an ellipse. Each particle exerts a force proportional to \( 1/r^2 \) on the other particles, where \( r \) is the distance between the individual particles. The strengths of the particles should be found such that the normal force on each particle is equal to the arbitrarily chosen value 1. The lay out of this test case is shown in Figure 3.7.

For modelling purposed the boundary condition points were distributed on a second ellipse inward of the singularity points ellipse. This is because the velocity of a singularity point is not defined at its center. However this does not affect the validity of the analysis. The initial guess fed into the GMRES algorithm was consistently taken as the zero solution. It was found from experience that a restarting factor \( m \) of 10 yielded the shortest numerical times, therefore this value was used throughout the following analysis.

In Figure 3.8 the evolution of the error is shown with respect to time, also the total direct time is shown for reference. This was done using an FMM with a \( 10^{-15} \) and \( 10^{-9} \) respectively. One can see that for low panel numbers the FastBEM solution is always slower, denoted by relatively high residuals at the break-even time with the direct solution. Increasing the panel count shows the potential of the FastBEM to achieve acceptable residuals faster than the direct method. However, high panel counts fail to converge to lower residuals.

Decreasing the FMM algorithm generally shifts the convergence curves to the left. Meaning less time is needed to achieve the same residuals. Note that this is only the case for residuals which are higher or in the same order of the FMM precision. If one wants to pursue very low residuals, the uncertainty in the FMM will cause stagnation of the residual.

As shown in subsection 3.2.4, preconditioning the system can promote the GMRES convergence for the high panel counts. Note that this will not work for very low residuals which originate from the FMM. Unfortunately, the preconditioning requires extra calculation time. The ideal preconditioner, being the LU factorization, requires for example the calculation of the influence matrix directly. Hence this will never exceed the performance of the direct method.
3.3 FastBEM

In Figure 3.9 the diagonal of the matrix was used as a preconditioner. The calculation time was added to the time of the initial step, this consists of \( N \) operations so only has a minor effect. It can be seen that the cases with the larger panel counts now also reach lower residuals. However they still stagnate at \( 10^{-9} \). Increasing the preconditioner to contain two super- and two subdiagonals did not improve this behaviour. Surprisingly increasing the FMM precision did not lower this virtual limit either.

However these residuals are considered low enough for its application in a panel code. The main conclusion is that the FastBEM is converging to some extent during first iterations and it does not become unstable, which is a crucial property for the panel code. If one wants to pursue machine precision residuals, a thorough mathematical analysis of the coupling between FMM and GMRES should be done. This should shed more light on the exact effect of the FMM precision, preconditioning and the original influence matrix on the convergence properties. This is considered to be outside the scope of this thesis.
**Figure 3.9:** Calculation time for the FastBEM algorithm using the diagonal of the influence matrix as preconditioner.
Chapter 4

Fast Panel Code

In previous section it was established that Fast algorithms exist to reduce the order of \( N \) with which the numerical time scales. The FMM can directly be used in the velocity calculations at one point due to all singularities in the flow. However since the \( A \) matrix requires the influence of all panels on each collocation point, the advantage of the FMM can not directly be used here. That is why the FastBEM method was discussed to avoid the direct calculation of the influence matrix \( A \).

Now the possibility of coupling this FastBEM with the panel code will be examined. As multiple integration approaches of the FastBEM in the panel code are possible, a computational time study should be carried out to select the right method for the validation case.

4.1 Methods

The FastBEM can be coupled to the panel code in different ways. The most straightforward one is to iteratively solve the whole system as shown in Figure 4.1. This option does not require building an influence matrix at all. The FMM calculates the induced velocity for an initial guess, after which this guess is constantly updated in the GMRES algorithm. Note that when iterating the wake, the initial guess can be set to the singularity strengths which were obtained in the previous wake iteration.

In the second option, the first iteration is calculated directly, meaning that influence matrices for the body and the wake are set up and saved. The obtained exact solution will is fed into the iterative wake procedure, which makes use of the FMM. Then the iterative solver is used to find the new circulation strengths of the system with the updated wake. Instead of solving the whole system, the body induced velocities which are a function of the vorticity strength are loaded from the direct calculation. The FMM is used to calculate the wake induced velocities on each collocation point. The sum of these contributions is fed into the GMRES algorithm. This yields new vorticity strengths, which can again be used to displace the wake. This procedure is visualised in Figure 4.2.
Note that the iterative solver is not required to converge to a specified (small) residual norm. It was established that the first steps of a FastBEM solver are generally converging. This means only a few FastBEM steps can be done, followed by a wake iteration. Since both the FastBEM and the wake are converging, the whole system is convergent. This saves computational time since FastBEM is not pursuing very low residuals each wake iteration.

Both methods have advantages and disadvantages. A nice feature of option 1 is that the direct calculations are omitted completely for the body. This allows for larger panel numbers than the second option since the first iteration of option 2 still scales with $O(N^2)$. 
However, the complete system enters the iterative solver each wake iteration which results in slower iteration steps.

An advantage of option 2 is that body-on-body influences can be saved. These do not change for a wing under different angles of attack for example. This means a batch of cases can be run in an efficient way. This is also an interesting property when dealing with solid boundaries such as wind tunnels. The tunnel influence on itself can be calculated once, independent of the test object in the tunnel.

4.2 Numerical study

In the previous section it became clear that 2 options can be chosen to couple the FastBEM algorithm with a panel code. A numerical study is done to determine which option is computationally most efficient. Note that the wake displacement algorithm is left out of the analysis since this will be the same for all configurations.

4.2.1 Scaling factors

In the direct method, the \( A_{\text{Basic}} \) matrix scales with \( N_b^2 \), where \( N_b \) denotes the number of body panels. These body-on-body interactions are only calculated in the first iteration and saved for later use. To satisfy the Kutta condition, the influence of the wake panels on the body panels must be taken into account. Computing the matrix \( A_{\text{Kutta}} \) scales with \( N_b \cdot N_w \), where \( N_w \) is the number of panels in the wake. So:

\[
\text{Direct method} \\
\begin{align*}
\text{First iteration:} & \quad \propto O(N_b^2) + O(N_b \cdot N_w) \\
\text{Following iterations:} & \quad \propto O(N_b \cdot N_w)
\end{align*}
\]

(4.1)

The FastBEM requires 2 FMM routines each iteration of GMRES. From subsection 3.1.3 it could be seen that this depends on the precision. Taking the scaling of Equation 3.10:

\[
\text{FastBEM on complete system (iprec 3)} \\
\begin{align*}
\text{Each iteration:} & \quad \propto n_{\text{iter}} \cdot 2 \cdot O(N_b^0.51 \cdot (N_b + N_w)^{0.61})
\end{align*}
\]

(4.2)

Since the developed code is steady, the body-on-body influences remain unchanged. Therefore it is possible to calculate it only calculate them the first wake iteration and use them later in the matrix multiplication \( A_{\text{Basic}} \cdot \Gamma \) in GMRES. The latter will require \( O(N_b) \) operations. Then the FMM requires the velocity due to \( N_w \) singularity panels at \( N_b \) target panels. So for \( \text{iprec 3} \):

\[
\text{FastBEM on wake (iprec 3)} \\
\begin{align*}
\text{First iteration:} & \quad \propto O(N_b^2) + O(N_b \cdot N_w) \quad (\text{direct}) \\
\text{Following iterations:} & \quad \propto n_{\text{iter}} \cdot 2 \cdot \left( O(N_b) + O(N_b^{0.51} \cdot N_w^{0.61}) \right)
\end{align*}
\]

(4.3)
4.2.2 Case study

Setting up explicit relations for the computation time for each method is not straightforward. It does not only depend on the number of panels in body and wake, but also on the number of wake iterations, convergence properties of GMRES and the precision of the FMM. The geometry and wake dynamics are also factors which influence the numerical efficiency.

To give an idea when which method is useful, a standalone wing at an angle of attack of 5 degrees is considered for different panel numbers. Convergence is not explicitly taken into account in the analysis due to the unpredictability of the GMRES algorithm at this stage.

Using the analysis of the previous section, the FMM time estimation relations obtained in Equation 3.9 and other numerical experiments, an assessment of the computational efficiency of the different methods can be conducted. In Figure 4.3 the computing time for varying body panel counts $N_b$ and wake panel counts $N_w$ are shown for the 3 cases.

From the combined timings one can see that the direct method is always the fastest for low panel counts as expected. This originates from the prefactor which is associated with the FMM. Then the FastBEM on the wake has the lowest computational time up to $10^6$ body panels. When extrapolating the trends one can see that the FastBEM on the complete system will perform better for very high panel counts. The reason for its
4.3 Outcome

rather poor performance w.r.t. wake FastBEM for low panel counts is due to the slower iterations within the GMRES algorithm.

When looking at the top view of Figure 4.3d, one can map the most efficient methods depending on the panel distribution, this is shown in Figure 4.4. A varying number of wake iterations are considered.

If the wake is not iterated, the direct method and the FastBEM method are the same, as confirmed by Figure 4.4a. The iterative solver for the whole system is more efficient for high panel counts whereas for more wake iterations, the wake FastBEM seems to be more profitable.

4.3 Outcome

In this section two coupling methods between the FastBEM algorithm and the panel method were proposed. One had the complete system (body + wake) as an input to the iterative solver. The other option was to calculate the first wake iteration directly, and only use the wake in the iterative solver. The latter proved to be most efficient when having more wake iterations for panel counts up to $O(10^5)$. Since the panel code
to be developed should deal with complex flows and actuation devices, accurate wake representations are needed, hence large wake iteration counts.

The case study also assumed a fixed amount of solver iterations for both methods. It can be expected, if one has to iterate the solution for the whole system, more iterations might be needed, thus further slowing down the complete FastBEM method.

Additionally, the wake FastBEM implementation allows to save the body-on-body influence matrix $A_{\text{Basic}}$ to treat similar cases. For an airfoil, varying angles of attack can be analysed with one $A_{\text{Basic}}$. Also the self influence of a wind tunnel can be saved independently of the test object. For these reasons the wake FastBEM implementation will be used in the Fast panel code.
Chapter  5

Verification and Validation

5.1 The actuator disk

The actuator disk concept is a method used to represent devices which extract or add kinetic energy from/to the flow. Therefore it is widely used for the initial design of propellers and wind turbines. It allows for the modelling of the aerodynamic behaviour without knowing specific turbine design parameters. The concept relies on a circular disk which exerts an uniform force on the flow, the so-called thrust force. At the edges, a tube shaped wake is shed due to the discontinuity in the force.

Since the flow is accelerated or decelerated and the mass flow rate is conserved in the stream tube, the wake will change diameter compared to the actuator disk diameter, as shown in Figure 5.1. In other words, the wake will align with the local flow field such that no normal flow acts through the wake. Since no body has to be modelled, it is possible to isolate the wake alignment procedure, making it a nice initial validation case.

\[\text{Figure 5.1: Actuator disk model for energy extracting device [24]}\]

As described above, the actuator disk can be represented by a tubular wake. In order to have vorticity in tangential direction, one should impose an increasing dipole strength
downstream. The flow field due to the shed vorticity is used to generate a force free wake, with a prescribed relaxation. First the wake convergence and the affecting parameters will be discussed. Then verification is done by comparing results to the 1D momentum theory using a prescribed thrust. Then the mass conservation is verified and finally a time analysis is established to show the advantage of the Fast panel code.

5.1.1 Wake convergence

The dipole strengths are chosen as arbitrary, but monotonously increasing values which results in a deceleration of the oncoming flow. Then the iterative wake procedure is executed of which some instances are shown in Figure 5.2. The colors represent the magnitude of normal velocity going through the wake, which is the basis for the calculation of the so-called wake residual.

Note that actual inviscid wakes tend to infinity and have was limited to 15 diameters downstream in this analysis. Furthermore the iterative wake procedure is done up to 5 diameters, behind which the wake displacement of the last panel is just extended to the panels downstream. These panels do not exactly line up with the local flow field, meaning there is a small normal components. However, the position of these far downstream panels does not affect disk parameters and can thus be ignored for the wake residuals.

One can observe an expanding wake with decreasing normal velocities through the wake each wake iteration. These normal velocities can be distilled in one number, the wake residual $r_w$ by taking the area weighted root mean square of the squared normal velocities

Figure 5.2: Evolution of normal velocity through wake with wake iterations

(a) 0 wake iterations
(b) 1 wake iteration
(c) 5 wake iterations
(d) 10 wake iterations
5.1 The actuator disk

as discussed before in section 2.3:

\[ r_w = \sqrt{\left( \sum_{i=1}^{N} \left( S_i \cdot V_{n,i}^2 \right) \right)^2} \]
\[ \sum_{i=1}^{N} (S_i) \]  

(5.1)

Now the effect of relaxation factor, FMM precision and mesh refinement on the wake residual is researched. In Figure 5.3 the wake residual each iteration can be seen for varying relaxation factors \( \beta \) and a FMM accuracy of \( 10^{-9} \). One can observe a quick convergence to a wake residual of \( O(10^{-8}) \). Other relaxation factors also level off at this value.

![Figure 5.3: Wake residual for different relaxation factors (FMM accuracy 10^{-9})](image)

It is possible that the change in displacement induces a velocity change below the precision of the FMM. However, it was found that changing the precision, or applying the direct method did not change the converged wake residual. Therefore one can look at the induced velocities used to displace the wake each iteration. This was quantified by a solver residual, which is an area weighted root mean square of the velocity used to displace the wake. In Figure 5.4 the evolution of this solver residual can be seen for each iteration. One can observe that after reaching a minimum, induced velocities are increasing again. Wake displacements become unstable, and when they are significantly high, they will eventually increase the wake residual again, which is omitted for clarity in Figure 5.3.

When reaching displacements with a magnitude of \( O(10^{-10}) \), numerical noise can trigger instabilities. The code is namely operating at or below the accuracy of the FMM, yielding random displacements which can be amplified. Another source of numerical noise which cannot be neglected is due to the numerical operations with floating point numbers. The accuracy after numerical mathematical operations can decrease compared to the floating point accuracy [25]. Another reason is the ill-posedness of a vortex sheet, which was established by Caflisch and Orellana [26], leading to inherent amplifications of any disturbances. The unstable effects of vortex sheets were used in the 1930s to model waves [3], as shown in Figure 5.5. A sinusoidal disturbance is introduced to a vortex sheet at \( t = 0 \), which grows and deforms in time due to the self influence of the wake.
Now it is established that resonant modes in the vortex sheets can be triggered when the solver residual stagnates. The reason for the stagnation of the wake residual however primarily lies in the spatial discretization. When plotting the normal velocity through the wake in the converged state, one can see a larger flux through the wake in the vicinity of the disk. There flow undergoes a rather sudden change, meaning the streamlines change direction. In these areas it is difficult for relatively large panels to follow the local flow smoothly. There are small oscillations because the panel tries to align with the streamline, and those oscillations cannot be overcome by changing the relaxation factor due to the nature of the problem. This means a refinement of the mesh is needed, especially when induction is high due to the large expansion near the actuator. When refining from 3500 to 11500 panels for the current problem, one can observe a decrease of the normal velocity through the wake near the disk as seen in Figure 5.6 for the same number of iterations and relaxation factor.

So it can be concluded that for a finer mesh the wake residual can reach lower values as shown in Figure 5.7. The solver residual is approximately in the same order, and is
fluctuating more for the fine panelling. This was expected since the fine panelling will capture high frequency noise, which are instabilities with small wavelengths. These are not captured with coarse panelling, so figuratively speaking more instability energy is captured when refining the mesh. This also means that continuous refinement will not lower the wake residual infinitely. Then numerical noise avoids further convergence, while the spatial discretization is fine enough to capture the flow.

It can be concluded that the stranding of the solver residual is due to the spatial discretization in the disk region, which sets a limit to the convergence of the wake residual. In this converged state, small oscillations can occur and are amplified, eventually leading to increasing wake residuals again. This means the implemented iterative wake procedure is stable when the user specifies a relative convergence criterion. When it stagnates, it is converged.

If the wake residual is not deemed low enough at this point, a finer mesh should be used. Ideally a non-uniform mesh, with a high panel density near the disk, should be used. This will be less sensitive to numerical noise due to the varying wavelength it captures. The refinement can be done to the numerical limits, where the precision of the FMM or the floating point accuracies avoids further convergence of the residual.
5.1.2 Verification using 1-D momentum theory

Actuator disks are popular design tools due to their straightforward theoretical solutions. From 1-D momentum theory for an ideal turbine it was found that the end velocity in the wake and at the disk can be defined in terms of an induction factor \( a \), as seen in Equation 5.2 and 5.3.

\[
    u_e = (1 - a) V_0 \quad (5.2)
\]

\[
    u_d = (1 - a) V_0 \quad (5.3)
\]

In these equations \( u_d \) and \( u_w \) represent the velocity at the disk and in the downstream wake respectively. The induction is a primary performance indicator for wind turbines since it directly relates to the power coefficient \( C_p \) and thrust coefficient \( C_T \) as described in [27], as shown in Equation 5.4 and 5.5.

\[
    C_T = 4a(1 - a) \quad (5.4)
\]

\[
    C_p = 4a(1 - a)^2 \quad (5.5)
\]

The latter is maximized when the induction \( a = 1/3 \), leading to a \( C_p \) of 16/27 and a \( C_T \) of 8/9. This thrust coefficient is the thrust \( T \) normalised by \( \frac{1}{2} \rho V_0^2 S_d \), with \( S_d \) begin the area of the rotor disk. Since the induction comes directly from the vorticity shed in the wake, it is useful to verify the induction for a specified \( C_T \). However, only wake vorticity is an input to the actuator model so \( C_T \) and vorticity \( \omega \) should be related. In Appendix A the vorticity equation is applied to a 2D actuator and then related to the induction factor. The resulting 3D formulation for the tangential vorticity is shown in Equation 5.6.

\[
    \omega_t = -\frac{1}{2} \frac{C_T V_0}{V_r} dx \quad (5.6)
\]

Variable \( dx \) represents the initial downstream length of the panel over which vorticity is distributed. Velocity \( V_r \) is the velocity at which vorticity is released from the actuator. The thrust coefficient is now set to 8/9, corresponding to an ideal rotor. The corresponding velocity field for this vorticity distribution is shown in Figure 5.8. Since the problem is axisymmetric, the out of plane velocities are 0, or \( \mathcal{O}(10^{-11}) \) for the numerical results.
5.1 The actuator disk

From the in plane velocity of Figure 5.8 it can be observed that the velocity downstream stabilizes approximately two diameters behind the disk. The velocity along the centerline is plotted in Figure 5.9, showing a converging end velocity. The end velocity is fairly independent of the mesh refinement with a maximum relative difference of 1%. An induction of $1/3$ corresponds to a theoretical non-dimensional end velocity of $1/3$, indicated by the dashed line.

However it can be seen that the end velocity is higher than the theoretical value. This is because 1D momentum theory assumes constant velocity across the cross-section. When plotting the velocity profile at the disk and further downstream shown in Figure 5.10, one can conclude this is not the case. However, average values correspond to the theoretical values, indicating a correct implementation of the actuator disk.
A final indicator is the power coefficient, which should correspond to the Betz limit of 16/27. It is calculated by taking the product of the prescribed thrust force and the average velocity at the disk. In Table 5.1 the $C_p$ values are shown for the different mesh refinements of Figure 5.9, with respect to the theoretical limit.

### Table 5.1: Verification of power coefficient w.r.t. Betz limit

<table>
<thead>
<tr>
<th>Number of panels</th>
<th>$C_p$ [−]</th>
<th>Relative difference %</th>
</tr>
</thead>
<tbody>
<tr>
<td>25000</td>
<td>0.5896</td>
<td>0.50%</td>
</tr>
<tr>
<td>50000</td>
<td>0.5905</td>
<td>0.35%</td>
</tr>
<tr>
<td>100000</td>
<td>0.5911</td>
<td>0.25%</td>
</tr>
</tbody>
</table>

#### 5.1.3 Verification of mass convergence

The reason for the expanding wake is to conserve the mass flux while the flow is slowed down. This mass conservation can be verified by calculating the cross section velocities downstream and integrating them over the cross sectional area. In Figure 5.11 the mass flux fluctuation is plotted in percent with respect to the mass flux at the disk. This is done for 3 mesh refinements. As discussed before, the expansion near the actuator is hard to capture with coarse meshes, evidently leading to larger fluctuations in mass flux. Refining the mesh improves this effect considerably, reducing the maximum difference in mass flux from 2% to 0.2%. For the refined case the average mass flux fluctuation is 0.002%, this corresponds to a wake residual of $O(10^{-11})$. Refining more does not lower these quantities any further due to numerical noise. However this is considered as an adequate mass convergence given a mass flow through the wake boundary of 0.4% by a reference solver [28].

![Velocity profile graph](image)

*Figure 5.10: Velocity profile at the disk and five diameters downstream*
5.1 The actuator disk

5.1.4 Time analysis

The goal of this Fast panel code was to gain significant calculation time benefits for a large number of panels. Since there are no body panels associated with the actuator disk, no linear set of equation must be set up. Therefore only the FMM will be used for the wake displacement. Since for the latter only velocities are needed at various points with a fixed singularity lay-out, the use of FMM is promising.

It was observed before that the FMM did not affect the wake convergence. Except if one pursues very low wake residuals, then uncertainties in the FMM might cause a stagnating wake residual. However in this case it is assumed that the latter is not the case, so any acceleration of the calculations leads to a time benefit when reaching the converged wake residual. In Figure 5.12 the time per iteration is displayed for varying panel counts. Similar to Figure 3.3, there is no clear advantage of using the FMM for low panel numbers. However, the slope of the direct method is significantly larger than for the FMM cases. For 60000 panels for example, the direct method is 30 times slower than the FMM with a precision of $10^{-9}$! Extrapolating the direct method to a panel count of 100000, then the Fast panel code would be roughly 100 times faster for this FMM precision.

Furthermore it was observed that the time estimates of Equation 3.9 and 3.10 gave the right order of time, but generally underestimated the needed time. Those relations were set up with random distribution of singularity points, which affects the amount of clustering compared to a fixed geometry. It is hard to generate accurate time estimates taking into account the geometry. Therefore it should be kept in mind that those relations only indicate the order of magnitude of the calculation times.

Finally it is worth to mention that 95% of the wake displacement procedure with FMM (iprec 3) is the actual time to calculate these velocities. The remaining 5% is used to calculate the normal component and displace the wake accordingly.
5.2 Impermeable wall

An impermeable wall is a building block useful in a series of aerodynamic applications. They can for example be used in the modelling of a wind tunnel, diffusers, nozzles, the ground and so on. Furthermore it is a useful verification case at this stage due to the explicit definition of the impermeable surface. For the actuator this zero normal flow through the wake was a result of aligning the wake with the local flow field. Now a linear set of equations is set up to solve for the singularity strengths of the surface. The induced velocity by these wall singularities should counteract the free stream velocity in normal direction at the collocation point. A good way to verify this is a nozzle, which is a tube with decreasing diameter, hence accelerating the subsonic flow to conserve mass.

5.2.1 Analytical solution

The analytical solution is based on the mass conservation, meaning \( \rho V S \) should be constant at each downstream section. So comparing two sections in the nozzle:

\[
\rho V_1 r_1^2 \pi = \rho V_2 r_2^2 \pi
\]

(5.7)

Here \( r \) is the local radius and \( V \) the local velocity. So the velocity ratio depends on the ratio of the local radii as:

\[
\frac{V_1}{V_2} = \frac{r_2^2}{r_1^2}
\]

(5.8)
5.2.2 Numerical results

The modelled nozzle has an inlet radius which is twice the exit diameter, with the radius following a quadratic curvature. The geometry and the panelling is shown in Figure 5.13.

The normal velocities at each collocation point is $O(10^{-10})$ when calculated with the FMM. Direct evaluation gives normal velocities of $O(10^{-15})$, which is machine precision as expected. This does however not imply that no flow is going through the walls at all. In the vicinity of panel edges for example, this boundary condition is not defined explicitly and cross flow might occur. In Figure 5.14 the in-plane velocities are shown at the center of the nozzle. Near the singularities, high velocities can be observed which follow from the inverse proportionality between induced speed and distance.

To assess how these velocities affect the aerodynamic characteristics of the nozzle, the integrated velocities are compared to the theoretical ones. In the analytical derivation it was shown that the integrated velocities should scale with the square of the radius.
ratio. This is shown by the red dashed line in Figure 5.15, with the reference being the integrated disk velocity at the center for the highest panel count.

![Figure 5.15: Integrated mean velocity along the nozzle](image)

It can be seen that for low panel counts, both the in- and outlet velocities are offset w.r.t. the theoretical solution. This is due to the large vorticity gradients near the edges. When refining this vorticity is spread over more panels, leading to smaller vortex strengths. This means these non-physically high velocities are more confined to the panel surface, leading to a better approximation of the in- and outlet velocities.

It was observed that the spatial discretization can yield high velocities locally. Despite the boundary condition still being satisfied, cross flow can occur on the panels away from the collocation points. A good measure to estimate this leakage is the evaluation of the mass flux at various sections along the nozzle. This is done for multiple panel densities and shown in Figure 5.16. One can see that for the fine panelling, this mass flow variation is generally within 5% compared to the inlet velocity. A convenient property of modelling solid walls with a dipole sheet is that wall leakage is not going one way. One dipole panel can lift the flow outside the nozzle due to the upstream vortex and move it back inside due to the downstream vortex of the panel. This means the mass flux will be constrained. The reason for the higher mass flux for lower panel counts is mainly due to the mismatch of the integrated inlet velocity.

![Figure 5.16: Relative mass flux w.r.t. the mass influx](image)
5.3 The 3D wing

The verification procedure is performed with increased complexity. In the actuator disk, only a wake was present due to a force discontinuity. Then it was proven that an impermeable boundary condition could be met. Now a body will be modelled that sheds a wake, with a vorticity depending on the body. Since this configuration models a body, it will reflect the correct implementation of the Kutta condition and body force calculations. 3D wings have received a lot of scientific attention, leading to theoretical formulations and experimental results which are commonly available. Pressure distributions and body forces can be compared and their sensitivity on meshing can be documented.

The iterative wake procedure will be challenged more due to the wake roll-up at the tips, which creates the need for a thorough convergence study of the wake.

5.3.1 Elliptical load distribution

Since Prandtl’s theoretical lifting line solution for the elliptic wing problem is straightforward and widely available [2], it forms a good basis for the verification of the linear system of equations and the calculation of the loads.

Analytical solution

The elliptic chord distribution for a wing with root chord $c_r$ and span $y$ is given by:

$$ c(y) = c_r \sqrt{1 - \left(\frac{2y}{b}\right)^2} \quad (5.9) $$

For a planar wing without sweep this yields the elliptic circulation distribution of Equation 5.10:

$$ \Gamma(y) = \Gamma_0 \sqrt{1 - \left(\frac{2y}{b}\right)^2} \quad (5.10) $$

This means that the lift distribution is also symmetrical through the Kutta-Joukowski theorem. Prandtl found that for this elliptical distribution, the downwash over the span is constant, leading to a simple expression for the induced angle of attack in Equation 5.11. This was the basis for Munk who established in 1921 that an elliptical distribution corresponds to the minimum induced drag for a planar wing of fixed total lift and span [29].

$$ \alpha_i = \frac{C_L}{\pi AR} \quad (5.11) $$
From above expression, the induced drag easily be found by Equation 5.12. One can see it is directly proportional to the square of the lift coefficient and inversely proportional to the aspect ratio.

\[ C_{D_i} = \frac{C_L^2}{\pi A R} \]  

(5.12)

The \( C_L \) depends on the \( c_l \) of the airfoil, the aspect ratio \( AR \), the angle of attack \( \alpha \) and the zero lift angle of attack of the wing \( \alpha_{0_w} \), as denoted in Equation 5.13.

\[ C_L = \frac{c_{l_{\alpha}}}{1 + \frac{\alpha}{\pi A R}} (\alpha - \alpha_{0_w}) \]  

(5.13)

For thin airfoil theory, the airfoil slope \( c_l \) is \( 2\pi \). Moreover circulation is related to the \( C_L \) through the Kutta-Joukowski theorem of Equation 1.10. Combining above expressions leads to the final formulation for \( \Gamma_0 \) in Equation 5.14 to determine the spanwise circulation distribution.

\[ \Gamma_0 = \frac{4b V_0}{AR + 2\alpha} \]  

(5.14)

This result is a useful benchmark for panel methods. However Prandtl’s lifting line solution does not take into account the chordwise variation of wake induced velocity. This chordwise variation of downwash is larger for low aspect ratios since the strong tip vorticity is relatively closer to the larger chord. This generally leads to an overestimation of the downwash, resulting in smaller lift and larger induced drag [30]. For low aspect ratio wings, Helmbold proposed a correction for the lift slope of the 3D wing w.r.t. the 2D airfoil [31]. Equation 5.13 becomes:

\[ C_L = \frac{c_{l_{\alpha}}}{\sqrt{1 + \frac{c_{l_{\alpha}}}{\pi A R} + \frac{c_{+\alpha}}{\pi A R}}} (\alpha - \alpha_{0_w}) \]  

(5.15)

**Numerical results**

Now Prandtl’s lifting line (LL.) solution is compared to the results of the fast panel code. A NACA0006 airfoil is distributed on an elliptical planform as shown in Figure 5.17.
In Figure 5.18 the circulation distribution can be seen for varying aspect ratios and a fixed angle of attack of 5 degrees.

An average relative difference of 0.1% was observed for the highest aspect ratio. Then increasing differences are associated with decreased aspect ratio. This was expected due to the limitations of Prandtl’s solution for low aspect ratio since chordwise variation of downwash is not taken into account. Helmbold made a semi-empirical correction for the lift curve slope of low aspect ratio wings, leading to the circulation distributions of Figure 5.19. One can see a better agreement for low aspect ratios whereas high aspect ratios deviate more.
It can be concluded that the fast panel code produces valid circulation distributions, even at intermediate aspect ratios (6-8) where lifting line solutions with or without corrections are inaccurate.

Since loads are derived from the pressure coefficient, it is useful to compare the pressure distribution at the center to the 2D airfoil properties. This is done in Figure 5.20, where it can be seen that higher aspect ratios approach the 2D solution as expected. However, to capture the suction peak one should increase the chordwise discretization, improving the total lift- and drag values.

The effect of the discretization of this suction peak is reflected in the lift curve slope of Figure 5.21a. Note that the deviation of the lifting line solution can again be ascribed to the varying downwash. The integrated drag values are out of the bounds of the inaccuracy of the lifting line solution. The nonzero drag at zero angle of attack also denotes these values are unreliable. Accurate $C_D$ estimations from surface pressure integration requires extremely dense panelling. Therefore a Trefftz analysis is done to estimate these drag values, which proves to accurately simulate the total drag, relatively independent of the discretization.

Figure 5.20: Chordwise pressure distribution for $\alpha = 5^\circ$

Figure 5.21: Lift- and drag coefficients for an elliptic wing with aspect ratio 8
This Trefftz analysis uses the far field velocities to calculate the induced drag. By applying the integral momentum in $x$ direction on a plane far downstream, one can relate the in-plane plane velocities to the induced drag. This method was proposed by Erich Trefftz, who found that the induced drag is the kinetic energy which is transferred into the crossflow [32]:

$$D = \frac{1}{2} \rho \int \int_{S_T} (v'^2 + w'^2) dS \quad (5.16)$$

In Figure 5.22 the Trefftz plane can be seen with its in-plane velocities. For the calculation of the induced drag this plane should be larger such that the whole perturbed flow field is captured. This region is defined as the region where these in-plane velocities are smaller than $10^{-6} \text{m/s}$, which for this case corresponds to a plane width of 4 and a height of 3 wingspans.

The downstream position is determined by checking the calculated drag values for various downstream instances. It was found that locating the Trefftz plane near the trailing edge overestimates the drag significantly, moving it downward gradually improves the estimation as expected. In Figure 5.23 the relative Trefftz drag difference at each $x$-position is shown. One can see that after 4 wingspans downstream, this variation is 0.05%, after which it drops. For the current stage of verification, this position is deemed far enough to accurately determine the drag. Therefore the Trefftz analysis in this report is done 4 spans downstream.
5.3.2 Wake roll-up

In Figure 5.24 one can see the iterative wake alignment for a standalone wing. From 10 iterations onwards, a clear tip vortex structure can be distinguished, leading to decreasing normal flow velocities through the whole wake.

The convergence of the normal velocities for different relaxation factors can be seen in Figure 5.25. Again one can observe that relaxation does not affect the minimum wake residual which is in the order of $O(10^{-4})$, after which it becomes unstable similar to the actuator disk.

The solver residual, based on the induced velocities to calculate the wake displacements is in the order of $O(10^{-4})$ for each case whereas this was $O(10^{-10})$ for the actuator disk. Therefore one cannot attribute these convergence instabilities to numerical noise coming
5.3 The 3D wing

Figure 5.25: Wake residual for different relaxation factors (FMM accuracy $10^{-9}$)

From floating point operations or FMM precision. In Figure 5.26 the wake residual is plotted separately for the core of the wake and the tips, considered to be the outer 15% of the span.

Figure 5.26: Wake residual for wake core and tips

It can be seen that the unstable wake residual is triggered by the tip roll-up. The unstable convergence behaviour of the tips is a known issue for panel codes without viscous correction. During the wake iteration process, vorticity panels can become very proximate of each other. This yields near singular induced velocities which require a lot of relaxation to create physical displacements. Otherwise vorticity panels may intersect with each other and even higher velocities are reached. This yields tip lay-outs as seen in Figure 5.27.

In actual flow these vorticity panels cannot be sustained either when in proximity of each other. As can be seen in Figure 5.28, vorticity panels create a large velocity gradients, thus shear stresses in the tip roll-up. In nature viscosity deals with these large shear
stresses, then vortex lines which are getting close would merge. Experimental research on this phenomenon was done by Schatzle [33], then Leonard [34] proposed a vortex filament surgery where reconnection is enforced for these panels. This is advisable when one wants to obtain a smooth tip wake geometry. However, since wake iterations are really fast, now the problem will partly be overcome with small relaxation factors and large amount of iterations.

5.3.3 Time analysis

In the actuator disk case, a significant time benefit was seen during the iterative wake procedure. This also leads to a severalfold acceleration of the wake iteration procedure in this case. This is not considered in the following time analysis to isolate the effects of the FastBEM implementation on the computing time.

The wake effects are iteratively solved using the FMM algorithm while the first wake iteration is still solved directly. The time to compute the latter is subtracted from the results since they are the same for both the direct and FastBEM method. Figure 5.29 shows the wake residual (excluding the tips) behaviour in time for the direct and FastBEM method for different FMM precisions. The configuration counts 4200 body panels and
11000 wake panels and had 3 FastBEM iterations per wake iterations. During these iterations, the solver residual typically reduced from $O(10^{-4})$ to $O(10^{-6})$.

![Figure 5.29: Core wake residual convergence in w.r.t. cumulative iteration time](image)

The quick solver convergence results in a small effect on the wake residual each wake iteration. Therefore each acceleration of the iteration directly translates into a time benefit. Thus for this case a 14% and 31% time benefit was obtained over 30 iterations for an FMM precision of $10^{-15}$ and $10^{-9}$ respectively.

### 5.4 Actuator in a wind tunnel

An important aspect of aerodynamic research are wind tunnel experiments. One of the main challenges is to assess how well the experiment resembles the real life case since flow is constrained within the wind tunnel walls. For wind turbines or propellers these blockage effects affect the performance. Glauert [35] made some corrections based on experiments of propellers in a wind tunnel with closed test section. This correction made use of one-dimensional axial momentum theory.

In 2002, Mikkelsen and Sørensen [36] developed a method to describe the induced wind speed through the rotor in a closed wind tunnel using one equation instead of a complete set. Later they generalized their method to apply it on open test sections [37]. Due to the general applicability and the straightforward analytical correction, this correction will be further analysed by using the Fast panel code.

#### 5.4.1 Method

In the correction using 1D momentum theory, a cylindrical wind tunnel cross section was assumed. However it is convenient to extend this correction to a square wind tunnel cross section. From the actuator analysis it was found that very fine meshing is needed to conserve mass throughout the actuator. Also Holt and Hunt [38] established that
very dense panelling is needed to avoid leakage through the wind tunnel walls. Therefore panels will be clustered near the actuator disk, which is shown in Figure 5.30.

Due to the dense panelling, the first iteration will be numerically expensive. However, the self influence of the tunnel only has to be calculated once and can be saved for later use. Once this is done, the tunnel can easily be added to any geometry without any significant added computational cost. For the actuator disk case, the \( C_T \) can be varied for example without having to set up the \( A_{\text{basic}} \). Furthermore, since no boundary condition is to be met on the actuator itself, its induced velocity can be added to the right hand side of the equation. Then one can solve the system of equations for the singularity strengths of the wind tunnel, to ensure a zero normal velocity at the collocation point.

### 5.4.2 Velocity at disk

For positive \( C_T \) values, the expansion of the wake is constrained due to the wind tunnel walls. This means the induction will be underestimated, thus a higher disk and end-velocity will be obtained. For negative \( C_T \) values the acceleration will be limited due to the walls, leading to underestimated velocities at the disk and downstream. This behaviour can be seen in Figure 5.31.

Note that the disk velocities are back calculated from the integrated mean velocities in the far wake. It was shown in the case of a nozzle, the integrated velocity is sensitive to the singularity strength gradients of the wall. Since these are largest near the actuator disk to counteract the induced velocity of the actuator, the integrated mean velocities are unreliable in this area.

Generally there is a good agreement between the proposed correction and the panel code results. Discrepancies can be seen for larger absolute \( C_T \) values. From the nozzle case it was observed that even though the boundary condition is satisfied in the collocation point, flow leakage can still occur. This means for that high induced normal velocities, which are associated with high \( C_T \) values, the wall is not completely impermeable and flow might leave the tunnel. This leads to a less constrained expansion, hence an underestimation of the tunnel effect. For negative \( C_T \) values, this would mean flow can be sucked into the tunnel, leading to ratio’s closer to one. This was not observed in this analysis. However, it is unsure if the proposed correction is valid for negative \( C_T \) values due to the lack of CFD reference data.
The cases should be run with a higher panel count in the tunnel since it was proven that this improves mass conservation. Furthermore, the downstream panelling of the actuator wake should also be increased. Panels are clustered near the actuator to have a good representation of the expansion. This means that a relatively coarse mesh is applied downstream. Since the velocity at the disk is calculated from a plane far downstream, these values might be unreliable.

5.4.3 Power coefficient

Since the disk velocities are directly related to the power coefficient, the thrust to power curve will be influenced when adding walls around an actuator disk. In Figure 5.32 the dashed red line represents the free stream $C_P$ corresponding to the prescribed $C_T$. One can see that wind tunnel walls can yield rotor performance which exceeds the Betz limit. Note that for the high $C_T$ values, the $C_P$ values in the tunnel will underestimated w.r.t. the CFD and correction case due to the lower disk velocity.

Furthermore tunnels with a square cross section are considered and shown in Figure 5.32. The wall interference effects for this cross sections are higher than the circular tunnel. In Table 5.2 the relative $C_P$ values are shown w.r.t. the Betz limit for $C_T = 8/9$.

<table>
<thead>
<tr>
<th>$R_t/R_r$</th>
<th>$C_P$ Circular tunnel [%]</th>
<th>$C_P$ Square tunnel [%]</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>+1.99</td>
<td>+5.18</td>
</tr>
<tr>
<td>2.4</td>
<td>+4.25</td>
<td>+6.48</td>
</tr>
<tr>
<td>2</td>
<td>+6.75</td>
<td>+9.08</td>
</tr>
</tbody>
</table>

The larger interference factors for square wind tunnels were also observed from experiments conducted by Theodorsen and Silverstein [39]. A vortex lattice method by Joppa [40] also confirmed this.
So despite the cross sectional area being slightly larger for the square tunnel and the average tunnel-to-wake distance being smaller, the tunnel effect is larger. This is probably due to the loss of axisymmetry which leads to other flow features in the corners.
6.1 Envisioned applications

It was shown that the developed Fast panel code is capable of rapidly evaluating complex 3D flows with actuation devices. Now a number of applications are listed which could efficiently be analysed by the panel code, possibly with minor modifications. Note that the applicability of the code is not restricted to the cases listed below.

**Aircraft with ducted fan propellers**

There is a growing interest for electric propulsion in general aviation. Often designs rely on ducted fan propellers, such as the Airbus E-Fan design shown in Figure 6.1. Due to the ability of the code to model wings, bodies and actuators, a complete inviscid model of the plane can be constructed. The rapid evaluation of the flow field makes it a valuable tool during early stages of the design.

![Airbus E-fan project](image)

*Figure 6.1: Airbus E-fan project [41]*

**Tidal turbine**

Multi-MW stream turbine generators gain popularity to extract energy from the tides. The Rotech Tidal Turbine as shown in Figure 6.2 has a large duct to increase the performance of the rotor. Not only the duct can be taken into account during the aerodynamic
Future developments

analysis, but also the seabed terrain. This code could therefore be used for siting purposed of these turbines.

Figure 6.2: Rotech Tidal Turbine [42]

Ground effect vehicles

The phenomenon of increased performance of a wing in proximity of an impermeable surface was already observed in the early stages of motorized flight. In the 1960s, highly efficient Ground Effect Vehicles (GEVs) were developed by the Soviet Union. These so-called ekranoplan [43] disappeared with the fall of the Soviet Union. Since then GEV’s were mainly designed for recreational purposes. In 2002, Boeing presented their Pelican concept, a cargo aircraft making use of the ground effect shown in Figure 6.3. This concept could be analyzed using the developed panel code. Adding an impermeable surface below the aircraft will alter the downwash, hence simulating this ground effect.

Figure 6.3: Boeing Pelican Ground Effect Vehicle [44]

6.2 Perspective for future use

The envisioned applications of previous section can basically be analysed with the current version of the code. But the future spectrum can be broader than those applications. Due to the rapid evaluation of the potential flow field, it might become a core module in a framework of potential flow codes to address other problems. A short introduction will be given to how this solver can be the basis for a viscous correction module, an aeroacoustic noise assessment and a static aeroelastic analysis.
6.2 Perspective for future use

6.2.1 Boundary layer coupling

When dealing with viscous flows, boundary layers emerge near the surface to account for a gradual velocity increase from 0 at the surface to the free stream velocity. The first codes taking this into account were the Eppler codes [45]. First the potential problem is solved to obtain the pressure distribution after which boundary layer parameters are calculated through momentum and energy equations. This proved to be inaccurate near separation, thus unreliable \( C_{L_{\text{max}}} \) estimations.

Then iterative procedures were introduced to account for the boundary layer in the potential solver. This is done by moving potential streamlines outward with the displacement thickness \( \delta^* \). This value denotes how much the free stream velocity \( U \) must be displaced from the surface to have an air transport equal to the case with a boundary layer. This is visualised in Figure 6.4.

![Figure 6.4: Definition of the displacement thickness [46]](image)

The displacement of the streamlines can be done geometrically by deforming the mesh. However this requires extra time and a new calculation of the influence matrices. Modifying the boundary condition using the so-called transpiration velocity \( w_T \) is therefore a better option. This is obtained by integrating the continuity equation across the boundary layer as done by Lighthill [47]:

\[
 w_T = \frac{1}{\rho} \frac{\partial}{\partial s} (\rho u_e \delta^*) \tag{6.1}
\]

Here \( s \) denotes the coordinate axis tangential to the body while \( u_e \) represents the local end velocity. The zero normal velocity boundary condition at the collocation point can now be replaced by the transpiration velocity to account for the boundary layer. A straightforward implementation of this is the weak interaction, as shown in Figure 6.5.

![Figure 6.5: Weak viscous-inviscid coupling](image)
This weak coupling would be straightforward to implement in the current version of the code. However, the code would be limited to attached flow since for separated flow the Goldstein singularity prohibits convergence [48]. Then a strong coupling is needed, such as the quasi-simultaneous method suggested by Veldman [48] and depicted in Figure 6.6.

\[ W_T \]

\[ U_e \]

**Figure 6.6: Strong viscous-inviscid coupling**

### 6.2.2 Aeroacoustic noise

The boundary layer is associated with a spectrum of noise depending on the local boundary layer properties [49]. The propagation of the sound waves emitted by the boundary layer can be described by Helmholtz equation. Fortunately, the used FMM library and the according FastBIE algorithm are able to deal with Helmholtz fields too, which could allow for a rapid evaluation of the sound fields around an object.

Monopole noise sources, as illustrated in Figure 6.7, can be added to each collocation point to mimic the aeroacoustic noise spectrum.

\[ \text{Figure 6.7: The pressure field of a 1kHz monopole source [50]} \]

At each collocation point, artificial sources with unknown strengths are added to account for the reflection of the body. Solving for these noise source strengths is similar to solving for the impermeable boundary condition. The influences from all unknown noise sources are put in the influence matrix while the summed contributions of the known sources are added to the right hand side. As such a linear system of equations with \( N \) unknowns is obtained.

This leads to a 3D representation of the sound field including the reflection and scattering due to the panels. Note that these sounds sources are additionally solved to the body
dipole strengths which ensure impermeability. Since boundary layer parameters change each wake iteration, the noise assessment can be done in the converged aerodynamic state. An interesting application would be the analysis of trailing edge serrations (shown in Figure 6.8), which can effectively reduce aerodynamic noise [51]. The attenuation is a function of the boundary layer parameters at the trailing edge and the dimensions of the serrations. Very fine meshes will be needed to capture the whole frequency spectrum accurately. Therefore it is believed that the presented Fast panel code can be an efficient tool for these purposes.

6.2.3 Static aeroelastic analysis

When looking at the wing of a flying glider, or the blades of a loaded wind turbine, one can observe they operate in a structurally deformed state. It can be convenient to take the interaction between the structural deformation and the aerodynamic properties into account in early stages of the design. For blade design this tip deflection is namely a design driver for the whole turbine [52].

As an initial structural coupling, one can represent the lifting surface by a beam with prescribed structural properties which is clamped at the center. When solving the aerodynamics, the loads can be used to deform this lifting surface. Which again can be used to solve the aerodynamic properties until a converged state is reached. The exact wake geometry is not required for each structural analysis so the wake can be updated in the same loop as the structural deformation.
Chapter 7

Conclusions and recommendations

The objective of the thesis was to develop a computationally efficient, accurate and robust implementation of a Fast inviscid panel code to model complex 3D flows with actuation surfaces. Now some concrete conclusions can be drawn with respect to the accuracy and efficiency of the model. There are still improvements to be made of which the most important ones are discussed in the recommendation section.

7.1 Conclusions

- First a standard steady panel code was developed which was able to solve potential flow fields around wings using an iterative wake procedure. From the implementation the following conclusions could be made:
  - Vortex rings and dipole panels are identical to represent a lifting body and can be interchanged in a problem. The direct evaluation of the latter requires more time than the former.
  - Adding sources with a prescribed strength such that the self influence counteracts the normal velocity of a panel was proven to improve the solution. The vorticity gradients were lower, leading to a higher accuracy.
  - The calculated singularity strengths can be used to determine the surface velocities by adding the different contributions of each panel. However, it was found that the gradient of the vorticity strengths should be added to minimize the errors associated with the discretization of the vortex sheet. Theoretically this is not needed when having high panel counts, but this limit was found to be impractically high.
  - The influence matrix could be split up in a body-on-body influence matrix $A_{body}$ and a wake-on-body influence matrix $A_{kutta}$. The body-on-body influence matrix contains the influences of all body singularities on the collocation points, which do not change during the wake iteration. This means $A_{body}$ only has to be

--
calculated once during the initial wake iteration. However the *wake-on-body influence matrix* does change due to the deforming wake and hence needs recalculation.

- The *body-on-body influence matrix* $A_{\text{basic}}$ can be stored and modified for different operating conditions. For the standalone wing for example, only the reference system should be altered when analyzing another angle of attack. Saving the influence matrix also proved useful in the analysis of wind tunnels, allowing to add different test objects without recalculating the self-influence of the tunnel.

- Since calculating the velocity at one point requires $N$ operations, with $N$ being the number of singularities in the flow, a Fast Multipole Method (FMM) was tested to accelerate these calculations:
  - The used FMM library [19] can evaluate the flow field around a quadrilateral dipole and source. The quadrilateral panel with strength $\Gamma$ is modelled by two triangles each having the strength $\Gamma$.
  - The evaluation of the mutual influence between $N$ panels is faster when $N \leq 600$ panels with respect to the direct method with an accuracy of $10^{-9}$.
  - The FMM proved extremely useful in the wake iteration procedure where velocity has to be calculated at each midpoint of the wake. For the actuator wake with 60000 panels, this speeded up this procedure with factor 30 for an accuracy of $10^{-9}$.

- The advantage of the FMM could not be used in the calculation of the *body-on-body influence matrix* $A_{\text{basic}}$ or the *wake-on-body influence matrix* $A_{\text{kutta}}$ since the influence of each panel on each collocation point is to be known. Therefore a FastBEM procedure was developed which combines an iterative solver with the FMM:
  - The GMRES algorithm proved to be an adequate iterative solver for linear systems associated with the panel code.
  - This solver typically converges the first iterations, except for high condition numbers. If the latter is the case, the system can be preconditioned to lower the condition number.
  - Applying the FastBEM on the whole influence matrix was deemed inefficient since $A_{\text{body}}$ is not defined explicitly anymore and this work should be redone each wake iteration. Therefore during the first wake iteration the influence matrices are calculated directly. Then $A_{\text{body}}$ is fed into the FastBEM procedure and the wake influence is evaluated through the FMM.
  - Since the wake is converging and the FastBEM is converging, one should not aim to pursue very low FastBEM residuals each wake iteration. A fixed amount of solver iterations can be set before iterating the wake, 3 solver iterations proved to be efficient in the wing case.
  - It was found that the FastBEM implementation did not significantly affect the wake residuals each iteration step. A slightly shorter iteration time was observed in the sample case, resulting in a 14% and 31% time benefit for an FMM precision of $10^{-15}$ and $10^{-9}$ respectively over a total of 30 wake iterations.
• Various test cases were run to check the validity of the iterative wake procedure, the linear set of equations and the post-processing.

  – An actuator disk was modelled by shedding a wake with a prescribed vorticity corresponding to a certain $C_T$. For a $C_T$ of $8/9$, the power coefficient was underestimated by $0.25\%$ while mass was conserved within the wake to a precision of $0.002\%$. For this a fine panelling near the actuator was needed to capture the expansion.

  – Then the impermeability boundary condition was tested by modelling a nozzle using dipole panels. The normal velocities at the collocation points were zero to FMM precision. However flow leakage was observed throughout the tunnel, this does not grow unbounded due to the dipole lay out on the surface. Refining improved the mass convergence and was around $5\%$ for 6000 panels.

  – The panel code is able to accurately solve for the circulation distribution of an elliptical wing. The according pressure distribution approaches the 2D case for high aspect ratios. To capture the pressure peak, a dense chordwise panelling (> 60 panels) should be applied. It was found that the force coefficients $C_L$ and $C_D$ obtained from the integrated pressure coefficients were not accurate. Even though this improves for fine panelling, $C_D$ values obtained from a Trefftz analysis are more reliable.

To have a physical tip roll-up, a very small relaxation factor was needed. Otherwise, the proximate vorticity panels generate near singular velocities and non-physical displacements consequently.

  – Finally an actuator in a wind tunnel was tested and compared to existing CFD data. Generally a good agreement was found between the two methods. Large positive $C_T$ values were associated with an underestimation of the tunnel effect, which can be assigned to flow leaving the tunnel due to the large expansion.

### 7.2 Recommendations

• During the testing of the standard panel code, it became clear that the force coefficients obtained from the integrated pressures were unreliable, especially in the tip region. A more robust way to obtain these coefficients might be through the Kutta-Joukowski theorem. The surface velocities and the local circulation can be used to extract a normal force.

• During the tip roll-up, proximate vorticity panels induce high velocities and hence large displacements on each other. Without very small relaxation factors, non-physical tip wake geometries were obtained. In reality, this region is governed by large shear stresses, and viscosity allows the merging of these vortex structures. Leonard [34] suggested a vortex filament surgery where reconnection is enforced for these panels to mimic these viscous effects. For more complex wake geometries, which can occur from wake interactions for example, small folds might occur. Then additionally the hairpin removal strategy as suggested by Chorin [53] can be employed. This would allow for less relaxation and faster convergence to the final wake configuration, increasing the overall computational efficiency of the code.
Conclusions and recommendations

During the analysis of the nozzle it was observed that flow leakage was hard to avoid, even for high panel counts. A higher order singularity panel might improve the impermeability over the whole surface. However, the FMM does not directly allow for these singularity panels. A closer look can be taken how these can be built with the available singularities. Possibly dipoles can be added pointing in the plane to have a varying strength along the plane. This would significantly improve impermeability, even for lower panel counts.

The FastBEM algorithm used in this code was mainly used as a black box. Therefore the exact effect of the FMM precision, preconditioning and the linear system of equations are unclear. During the FastBEM demonstrator case, residuals smaller than $10^{-9}$ were hard to obtain. A thorough mathematical analysis of the coupling between the FMM and the GMRES algorithm can shed light on these effects and enhance the predictability of the numerical time.

A drawback of the code is the direct evaluation of the body-on-body influence matrix $A_{\text{basic}}$, which scales with $O(N^2)$. It was found that this direct calculation can be overcome by employing a FastBEM method, which is essentially faster for high panel counts. However, since $A_{\text{basic}}$ is not formed explicitly, the FastBEM procedure on the whole system would be needed each wake iteration step, slowing the iterations down accordingly. A possible way to alleviate the computational load at the initial step is to directly evaluate the body-on-body influences on a coarser mesh. This matrix can be fed in the iterative solver which will now not only account for a change in the wake, but also for the discretization errors made in the initial calculation. This spreading of the numerical effort is likely to improve the global numerical efficiency.
References


REFERENCES


Appendices
Appendix A

$C_T - \omega_t$ relation for an actuator disk

Verification of the actuator disk is done by comparing the results to 1D momentum theory. Therefore a thrust coefficient $C_T$ is prescribed, defined as the thrust $T$ normalised by $\frac{1}{2} \rho V_0^2 S_d$, where $S_d$ is the area of the actuator disk. The thrust coefficient is directly related to the wake vorticity through the vorticity equation of Equation A.1. This formulation can be derived by taking the curl of the Navier-Stokes equation, as described by [54].

\[
\frac{D\vec{\omega}}{Dt} = \frac{\partial\vec{\omega}}{\partial t} + (\vec{u} \cdot \vec{\nabla})\vec{\omega} = (\vec{\omega} \cdot \vec{\nabla})\vec{u} + \frac{\mu}{\rho} \vec{\nabla}^2 \vec{\omega} + \frac{\vec{\nabla} \times \vec{f}}{\rho} \tag{A.1}
\]

For a steady, inviscid and axisymmetric problem, this equation is reduced to:

\[
\frac{D\vec{\omega}}{Dt} = (\vec{u} \cdot \vec{\nabla})\vec{\omega} = \frac{\vec{\nabla} \times \vec{f}}{\rho} \tag{A.2}
\]

For simplicity the 2D case will be discussed, meaning there is no velocity in $z$ direction, however vorticity in $z$ direction is possible. Rewriting equation Equation A.2 yields:

\[
\begin{align*}
\frac{D\omega_x}{Dt} &= u \frac{\partial \omega_x}{\partial x} + v \frac{\partial \omega_x}{\partial y} = \frac{\vec{\nabla} \times \vec{f}}{\rho} \cdot \vec{e}_x \\
\frac{D\omega_y}{Dt} &= u \frac{\partial \omega_y}{\partial x} + v \frac{\partial \omega_y}{\partial y} = \frac{\vec{\nabla} \times \vec{f}}{\rho} \cdot \vec{e}_y \\
\frac{D\omega_z}{Dt} &= u \frac{\partial \omega_z}{\partial x} + v \frac{\partial \omega_z}{\partial y} = \frac{\vec{\nabla} \times \vec{f}}{\rho} \cdot \vec{e}_z \tag{A.3}
\end{align*}
\]

The actuator disk has a uniform loading in $x$ direction, meaning $\vec{f} = f \cdot \vec{e}_x$. Then the curl of the force vector reads:

\[
\vec{\nabla} \times \vec{f} = \begin{vmatrix} \vec{e}_x & \vec{e}_y & \vec{e}_z \\ \frac{\partial}{\partial x} & \frac{\partial}{\partial y} & \frac{\partial}{\partial z} \\ f & 0 & 0 \end{vmatrix} = 0\vec{e}_x + \frac{\partial f}{\partial z} \vec{e}_y - \frac{\partial f}{\partial y} \vec{e}_z \tag{A.4}
\]
This means there is only vorticity in $z$ direction:

$$\frac{D\omega_z}{Dt} = \frac{D\omega_z}{Dx} \cdot \frac{Dx}{Dt} = -\frac{1}{\rho} \frac{\partial f}{\partial y}$$ (A.5)

Finally the vorticity jump over the disk can be written as:

$$\frac{D\omega_z}{Dx} = -\frac{1}{\rho u} \frac{\partial f}{\partial y}$$ (A.6)

Integrating yields the vorticity $\omega_z$ shed at the sides of the actuator:

$$\omega_z = -\frac{f}{\rho V_r}$$ (A.7)

Here $V_r$ denotes the velocity with which vorticity is released. With the distributed force $f = T/S_d$, where $S_d$ is the surface area of the actuator disk, one can finally write $\omega_z$ in function of the thrust coefficient $C_T$. Note that the free stream velocity $V_0$ is the velocity in $x$-direction $u$.

$$\omega_z = -\frac{1}{2} \frac{C_T V_0}{V_r}$$ (A.8)

This 2D analysis can easily be expanded to a 3D case, there is only a change in force $f$ along the radial direction, meaning shed vorticity is directed in the tangential direction. The expression for this tangential vorticity is shown in Equation A.9, with $dx$ being the panel length downstream.

$$\omega_t = -\frac{1}{2} \frac{C_T V_0}{V_r} dx$$ (A.9)