Surface Gradient Algorithms for Unstructured Grid Panel Methods

Thesis

by

Thomas Maarten Altena

Master of Science
in Mechanical Engineering

To be defended on
July 18^{th} 2024

UNIVERSITY OF TWENTE.

University of Twente
Faculty of Engineering & Technology
Department of Thermal & Fluid Engineering
Drienerlolaan 5 - 7522 NB Enschede

Thesis Committee
Chairman: Prof. Dr. Ir. C.H. Venner
Supervisor: Dr. Ir. A. van Garrel
External member: Dr. A.R. Thornton

Version of July 10, 2024

Contents

Summary

Advanced low-order panel methods based on the internal Dirichlet boundary conditions use piecewise constant source and dipole distributions to model inviscid, incompressible (potential) flow over complex geometries. In these formulations the perturbation velocity tangential to the surface in the nodes is obtained in a post-processing step from the surface gradient of the dipole distribution. The resulting flow velocities can be used to determine the aerodynamic loads. Determining the surface gradient for a structured grid panel method is relatively straightforward, while unstructured grid panel methods are challenging with respect to accuracy, computational cost, and data structure complexity. The benefit of a surface gradient method capable of also handling unstructured grids lies in the possibility to use automatic grid generators for complex geometries. This results in a big time-saving factor in grid-based numerical panel methods.

In this research, the focus lies on two numerical approaches to determine the surface gradient at the panel corner points (nodes):

1. Green-Gauss surface gradient reconstruction
2. Least-squares surface gradient reconstruction

In particular the Green-Gauss gradient reconstruction scheme is extensively tested to reconstruct the gradient of both linear and non-linear functions. This is done for structured and (highly) unstructured grids over flat and curved surfaces in 3D space. Additionally, to support the Green-Gauss gradient scheme, the half-edge data structure is implemented in Fortran to navigate efficiently through both structured and unstructured grids.

The tests revealed that linear functions can be represented exactly on flat surface grids while higher order functions are at least first order accurate. On equidistant structured grids, even quadratic functions can be represented exactly, while higher order functions are second order accurate. Unstructured grids where the nodes are perturbed result in the least accurate reconstruction of the surface gradient, but these are also at least first order accurate. Highly curved grids such as over the surface of a sphere reveal that the linear functions cannot be represented exactly anymore by the Green-Gauss method, but the gradient of all functions can be reconstructed with at least first order accuracy on unstructured meshes. The least squares method on the other hand, is able to exactly reconstruct the gradient of linear functions on curved grids.

1 Introduction

In this day and age, many industrial flow problems are simulated with the help of Computational Fluid Dynamics (CFD) solvers, such as Ansys Fluent, STAR-CCM+ and even open source software as in OpenFOAM. These problems often involve complex geometries and turbulent flows for which computational costs are high. However, simplifications with respect to the flow i.e. flows without vorticity in which effects of viscosity diminish (potential flow), result in the fact that much simpler computational methods can be used that drastically reduce the computational costs.

1.1 Numerical Panel Methods

In fluid dynamics, many valuable insights have resulted from observing the difference between fluid flows in real-life and their corresponding potential flows. Potential flows can be seen as more idealized flows, which are not subjected to the complex effects of vorticity. Due to the assumption of an irrotational flow in combination with an incompressible flow, the velocity field that is described by a scalar function satisfies Laplace's equation. The Laplace equation is known to be linear resulting in the fact that the principle of superposition is valid. This means that a summation of solutions to the Laplace equation also results in a solution to the Laplace equation, which makes it possible to solve complex problems by adding simple solutions. As a result, solution methods to potential flow problems are often used in engineering practices, such as the preliminary design of aerodynamic or marine structures [1]. Katz and Plotkin [2] discussed several analytical solution techniques that are only applicable by significantly simplifying the geometry, such as flat or zero thickness surfaces, and also applying the boundary conditions at these simplified geometries. However, numerical techniques, such as panel methods, are not restricted to significant simplifications in the geometry and can even treat realistic and complex geometries while also applying the boundary conditions on the actual surface. These panel methods divide the surface in panels on which singularity elements, such as sources and doublets are placed which are known to be a solution to Laplace's equation. In contrast to the finite volume method, on which many of the CFD solvers are based, these panel methods only determine the solution on the surfaces within the flow domain and not the entire flow domain which can result in a serious reduction of computation time.

1.2 Developments and Problems in Numerical Panel Methods

Although panel methods are a great tool for preliminary design regarding fast computations for complex structures, they are not capable of dealing with viscous, rotational and supersonic ($M > 1$) flows. Many applications of panel methods restrict themselves to high Reynolds number ($> 10^5$), subsonic flows over streamlined surfaces to distinguish fully-attached flows. Within these applications, the regions outside the boundary layer and the wake are considered, as neglecting the viscous effects is an appropriate assumption there [2]. Consequently, different authors developed methods to couple panel methods for the inviscid flows external to the boundary and wake regions with boundary layer approximation methods for the viscous effects as depicted in Figure (1). As one of the first, Drela [3] accomplished a combined computational method that couples an integral boundary layer method with a panel method for 2D airfoils [4]. This led to the development of his interactive program called XFOIL [5] which is still widely used for airfoil analysis and design. Consequently, this was extended to 3D constructions, such as the approach by Ramos García, Sørensen and Shen [6] that predicted the aerodynamic behaviour of wind turbine airfoils by a panel method for the inviscid part, while the viscous part is modeled by solving boundary-layer equations, laminar and turbulent, with extension for 3D rotational effects. Additionally, van Garrel [7] presented a coupling of a low-order time-stepping panel method with a 3D integral boundary layer method that could, even in the case of a separated flow, account for viscous effects. On the other hand, Ortega, Flores and Oñate [8] did not account for viscous effects, but they adopted a time-stepping 3D low-order panel method to analyse effects of small configuration changes along specified paths.

Many of these developments preceded the arrival of automatic mesh generation algorithms. Creating meshes for complex structures is a time-consuming process within computational fluid dynamics and meshing tools such as GMSH, MeshLab and Rhinoceros can create meshes within considerably less time. These automatic mesh generators often result in unstructured meshes and authors such as Pedro [9] and Satterwhite [10] discussed their effects on the accuracy and applicability to panel methods. Especially mesh navigation and the reconstruction of the surface velocity resulting from the unstructured low-order panel methods are the cause of increased complexity and a decrease in accuracy.

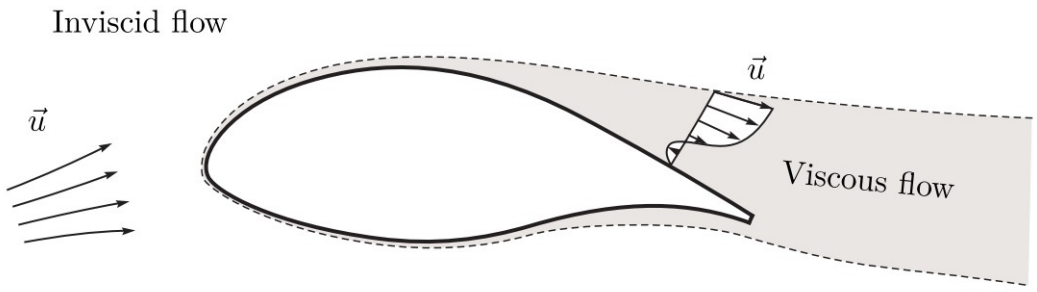


Figure 1: Division of the flow domain in an inviscid flow region outside of the boundary layer and wake regions and a viscous flow region which includes the boundary layer and wake regions [4].

1.3 Outline of the Thesis

Consequently, this thesis will analyse different surface gradient reconstruction methods in combination with a data structure to navigate unstructured meshes. The goal is to contribute to the development of a hybrid grid (low-order) panel method capable of handling both structured and unstructured grids. This thesis is structured as follows:

- In the chapter Problem Analysis and Specification, different surface gradient reconstruction schemes are analysed and discussed. Additionally, the difference between structured and unstructured grids are contemplated. Their corresponding secondary grids are also evaluated and the effects of grid quality are discussed. This chapter is closed by the evaluation of the requirements for a data structure and which data structures could be suitable.

- The following chapter considers the methods that are selected to solve for the surface gradient on (un)structured meshes. The half-edge data structure is further analysed as well as the Green-Gauss surface gradient reconstruction method and the least squares gradient reconstruction scheme. Finally, different test meshes and functions are discussed.

- The chapter Results and Discussion shows the results of the tests on flat and curved quadrilateral and triangular meshes. Both surface gradient reconstruction schemes are compared and show that the errors are between $\mathcal{O}(h^2)$ and $\mathcal{O}(h)$ for the graphs that converge. It is also shown that there is likely a mistake in the implementation with regards to the half-edge data structure or the Green-Gauss method.

- The last chapter provides suggestions for future work and concludes this research.

2 Problem Analysis and Specification

In Appendix (B), the potential flow problem is described. The velocity potential function Φ satisfies the Laplace equation for irrotational, inviscid, incompressible flows. Following the general solution procedure based on Green's second identity as discussed in Katz and Plotkin [2], definitions for the velocity potential, doublet strengths and source strengths are obtained. With the help of Dirichlet boundary conditions, it can be deduced that the surface gradient of the velocity potential can be determined with the surface gradient of the doublet strength (μ):

$$\vec{\nabla}_S \Phi = -\vec{\nabla}_S \mu. \tag{1}$$

Upon neglecting the wake, only the surface of interest has to be divided into flat panels. On these panels doublet elements are placed for which the strengths can be determined by solving a system of equations. These doublet strength values are assigned to the panel midpoints and with this discrete set of doublet strengths, the surface gradient of the velocity potential function must be determined on both structured or unstructured panelled surfaces.

2.1 Surface Gradient Reconstruction Schemes

Determining the surface gradient in an unstructured grid is not as straightforward as it is for structured grids. Overall, there are hardly any gradient schemes that are particularly used to determine the surface gradient, but there are several methods to determine the standard gradient which in turn can be used to determine the surface gradient. Apart from a few distinct reconstruction schemes, most schemes can be classified as a Green-Gauss gradient scheme or a least squares gradient scheme.

2.1.1 Green-Gauss Surface Gradient Reconstruction Scheme

The first gradient reconstruction scheme discussed here is the Green-Gauss scheme which is often used in the finite volume method. This scheme is used for both 3D cells and 2D cells where a cell is the term used for one mesh element e.g. a triangle or tetrahedral in 2D and 3D respectively. The Green-Gauss gradient scheme is based on the gradient theorem i.e.

$$\iiint_{V_C} \vec{\nabla} f \, dV = \iint_{S_C} f \bar{\nu} \, dS, \tag{2}$$

in which f denotes the quantity of which the gradient is sought, V_C denotes the volume of the cell, S_C is the enclosing surface and $\bar{\nu}$ denotes the unit normal at every point on S_C. The Green-Gauss method only uses polygons and polyhedrons of which the bounding surface can always be subdivided into $k = 1, 2..., F$ straight or planar faces. Additionally, the midpoint integration rule is used which states that the value at the centroid of each face or cell plus a second order correction term should be equal to the average of a quantity over that face or cell [11]. For a grid spacing h and for a centroid ($\vec{C}$) of a cell C, this can be written as:

$$\frac{1}{V_C} \iiint_{V_C} \vec{\nabla} f \, dV = \vec{\nabla} f(\vec{C}) + \vec{\mathcal{O}}(h^2). \tag{3}$$

In a similar fashion, this can be done for the face centroids. Subsequently, by combining these equations the 'final' form of the Green-Gauss gradient reconstruction scheme becomes:

$$\vec{\nabla} f(\vec{C}) = \frac{1}{V_C} \sum_{k=1}^{F} f(\vec{c}_k) S_k \bar{\nu}_k + \vec{\mathcal{O}}(h). \tag{4}$$

Here, S_k denotes the area of face k, $\bar{\nu}_k$ denotes the normal to face k and $\vec{c}_k$ denotes the centroid of face k. This equation is still exact, but dropping the final term results in a first order approximation [11], [12]. As mentioned before, this scheme is resultant from the finite volume method, but in this research, the surface gradient of the doublet strength is required and therefore, this equation can be written as:

$$\vec{\nabla}_S \mu(\vec{C}) = \frac{1}{S_C} \sum_{k=1}^{F} \bar{\nu}_k \mu(\vec{c}_k) L_k + \vec{\mathcal{O}}(h), \tag{5}$$

in which L_k denotes the k^{th} side of the contour with $\vec{c}_k$ as its center. This equation is derived and described in more detail in Section (3.2). Gradient reconstruction in the finite volume method is quite similar to the surface gradient reconstruction, but the reconstruction for the surface gradient only considers a curved surface in 3D space and not an entire volume as can be seen in Figure (2).

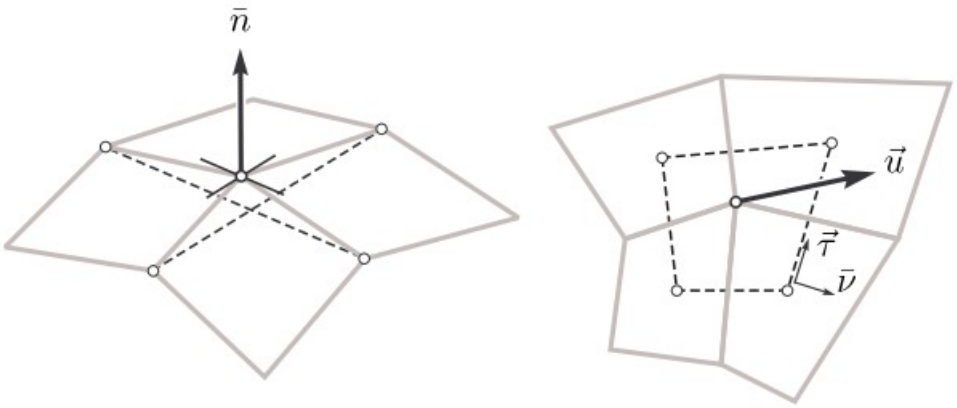


Figure 2: Curved surface in 3D space with definitions for the vertex normal $\bar{n}$, tangential contour vector $\vec{\tau}$ and contour normal $\bar{\nu}$ with a tangential velocity vector $\vec{u}$ along the surface [4].

So, the type of problem or method in which the Green-Gauss scheme is used is of importance here, because in cell centered methods, the values of a quantity are only known at the cell centroids and not at the face centroids. This induces the problem that $\mu(\vec{c}_k)$ needs to be obtained in terms of the values at the cell centroids. In Figure (3), the 2D schematic on the right shows the direction of the unit normal vectors $\bar{\nu}_k$ originating from the center of each of the cell's faces. As discussed previously, the midpoint integration rule requires this point to be at the center of each of the faces. As the right schematic in Figure (3) shows, these points are not always on the lines connecting the (blue) cell centers which takes away the possibility to use linear interpolation. Linear interpolation is known to be second order accurate and therefore it would not degrade the accuracy of the Green-Gauss method [11]. As a result, many authors researched this problem e.g. Sozer et al. [12] tested several face interpolation methods such as simple face averaging, inverse distance weighting, weighted tri-linear face interpolation and a weighted least squares variant. The first two variants are inconsistent and should therefore be avoided while the last two variants are first order accurate, though be it with significantly larger errors than a simple least squares method. Deka et al. [13] invented a so-called Modified Green-Gauss method which reconstructs the gradients using normal derivatives at the face resulting in a consistent first order accurate scheme. By constructing an auxiliary volume around a centroid, Athkuri and Eswaran [14] constructed the Circle Green-Gauss method which gives consistent first order accurate gradients at the centroids for triangular and tetrahedral meshes. In another research, Nishikawa [15] proposed an Implicit Green-Gauss variant that forms a globally coupled linear system of equations which can be solved efficiently with iterations. Similarly, Syrakos et al. [11] proposed a new method which used so-called outer iterations. All these schemes add computational cost or complexity to the 'standard' Green-Gauss method due to the fact that the value at the face centers are not known. These problems are coupled with, in most cases, the

finite volume method in which a stencil as shown on the left in Figure (3) is not possible. The stencil on the left in Figure (3) will denote the nodal variant of the Green-Gauss method and in contrast to the stencil in the right schematic of Figure (3), the centers of the faces always lie on the line between two cell centers as these are now the boundaries of the cells. As a result, the values $\mu_k(\vec{c}_k)$ can be determined as the average of the cell centers bounding that face. Only a few papers use or mention this scheme i.e. Katz and Sankaran [16] make use of this scheme and Syrakos et al. [11] mention that this scheme is at least first order accurate which is also concluded by the work of Katz and Sankaran. Finally, for boundary nodes, this scheme cannot be applied as it is not possible to form a closed contour around the node by connecting cell centers.

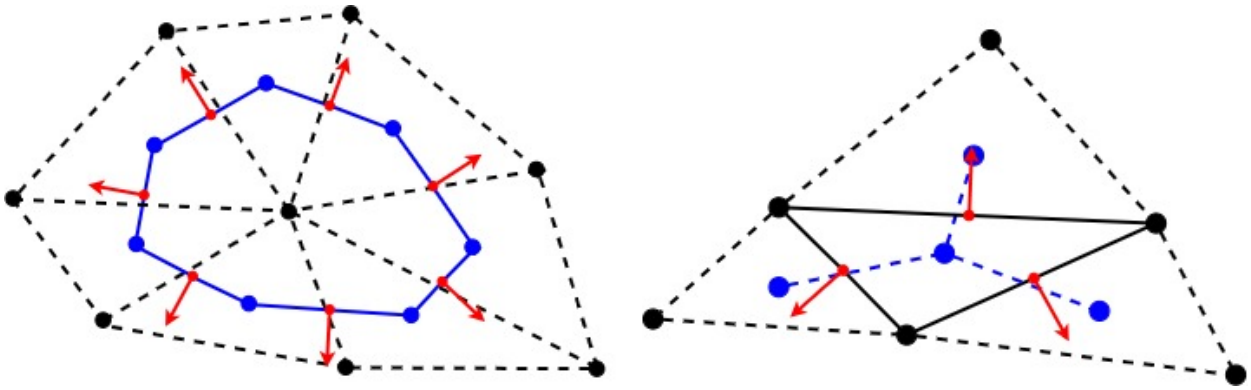


Figure 3: The Green-Gauss stencil for a dual cell (left) and a normal cell (right).

2.1.2 Least Squares Surface Gradient Reconstruction Scheme

Another big branch of gradient reconstruction schemes are the ones based on the least squares method. For a cell C, the gradient can be determined at its centroid $\vec{C}$, by including the neighbouring cell centroids of cells that share an edge (the nearest neighbour stencil) or by including the cell centroids of cells that share a node (the extended stencil). The right schematic in Figure (4) shows the nearest neighbour stencil with the black arrows, while the extended stencil also includes the red arrows. This directly shows the contrast with the Green-Gauss based methods, as it is now very simple to incorporate more data i.e. neighbouring centroids. An extended stencil can, in many cases, increase the accuracy of the solution, but this increases the computational cost of the problem. As for most applications the nearest neighbour stencil is already quite accurate, only for a few cases the extended stencil is required. Therefore, Seo et al. [17] examined the possibility of switching between these methods which resulted in a more economical method with a good accuracy that switches between the stencils based on the condition number of the least squares matrix. For now, both stencils are treated individually. So, assuming there are $k = 1, 2..., K$ neighbouring cell centroids to include, then at each of these neighbouring centroids ($\vec{N}_k$) the values of f are expressed as a Taylor series expansion about the centroid of $\vec{C}$. For all neighbours K, this can be written as:

$$f(\vec{N}_k) = f(\vec{C}) + \vec{\nabla} f(\vec{C}) \cdot (\vec{N}_k - \vec{C}) + \mathcal{O}\big(h^2\big), \tag{6}$$

where the Taylor series is terminated after the first derivative resulting in an $\mathcal{O}\big(h^2\big)$ term. Here, the vector $(\vec{N}_k - \vec{C})$ denotes the distance from the centroid of cell C to the centroid of each of its neighbours. For simplification purposes this distance vector will now be denoted as $\vec{d}_{Ck}$. Rewriting the equation above gives:

$$\vec{d}_{Ck} \cdot \vec{\nabla} f(\vec{C}) = f(\vec{N}_k) - f(\vec{C}). \tag{7}$$

Here, the $\mathcal{O}(h^2)$ terms are dropped from the equations which ensured the equations were exact. All K equations can be put into a single matrix equation which, if written as in the form above, is of the form $[A]\vec{x} = \vec{b}$:

$$\begin{bmatrix} d_{C1,x} & d_{C1,y} & d_{C1_z} \\ d_{C2,x} & d_{C2,y} & d_{C2_z} \\ \vdots & \vdots & \vdots \\ d_{CK,x} & d_{CK,y} & d_{CK_z} \end{bmatrix} \begin{Bmatrix} \frac{\partial f}{\partial x}(\vec{C}) \\ \frac{\partial f}{\partial y}(\vec{C}) \\ \frac{\partial f}{\partial z}(\vec{C}) \end{Bmatrix} = \begin{Bmatrix} f(\vec{N}_1) - f(\vec{C}) \\ f(\vec{N}_2) - f(\vec{C}) \\ \vdots \\ f(\vec{N}_K) - f(\vec{C}) \end{Bmatrix}. \tag{8}$$

In general, due to the fact that the $\mathcal{O}(h^2)$ terms are dropped, this system does not have a solution. More to the point, if $\vec{b}$ does not lie in the column space of $[A]$, then the unknowns ($\vec{x}$) of the linear system ($[A]\vec{x} = \vec{b}$) cannot be solved [11], [18]. To get the best result possible, the vector $\vec{x}$ that results in the smallest error must be found. Each equation has an error:

$$e_k = f(\vec{N}_k) - \left[f(\vec{C}) + \vec{d}_{Ck} \cdot \vec{\nabla} f(\vec{C}) \right]. \tag{9}$$

When the projection of the error onto the column space of $[A]$ is equal to zero, the error is minimised. If matrix $[A]$ has independent columns, the normal equations have a unique solution i.e. $\vec{x}$ is known as the least squares solution [11], [18]. First pre-multiply both sides of the linear system with the transposed of $[A]$:

$$[A]^T[A]\vec{x} = [A]^T\vec{b}. \tag{10}$$

Additionally, to isolate $\vec{x}$, pre-multiplying both equations with $([A]^T[A])^{-1}$ gives:

$$\vec{x} = ([A]^T[A])^{-1}[A]^T\vec{b}, \tag{11}$$

since a matrix multiplied with its inverse results in the identity matrix. This equation describes the least squares solution which, in this case, is an equation for the gradient. However, when looking at the schematics in Figure (4) showing the least squares stencils, it can be seen that the neighbours surrounding the point of interest are not distributed evenly. Additionally, some neighbours are much further away than others and might therefore be deemed as less influential. To account for this, Equation (7) can be weighted which gives:

$$w_k \vec{d}_{Ck} \cdot \vec{\nabla} f(\vec{C}) = w_k [f(\vec{N}_k) - f(\vec{C})], \tag{12}$$

where w_k denotes the weighting factor. Similar as before, the least squares solution can be determined, which is now given by:

$$\vec{x} = ([A]^T[W]^T[W][A])^{-1}[A]^T[W]^T[W]\vec{b}, \tag{13}$$

in which $[W]$ is a diagonal matrix with the weighting factors on its diagonal. As discussed previously, there are K equations and due to the definitions of the matrices $[A]$ and $[W]$, the matrix multiplication ($[A]^T[W]^T[W][A]$) always results in 3x3 matrix. So, as a simplification, the resulting 3x3 matrix will be denoted as $[G]$ i.e.

$$\vec{x} = [G]^{-1}[A]^T[W]^T[W]\vec{b}. \tag{14}$$

As will be discussed shortly, the weighting factor is often chosen as a grid dependent variable. This means that for stationary grids, the matrix $[G]$ can be determined beforehand which makes the least squares gradient efficient to compute [18]. With regards to the weighting vector, there are several options. Inverse distance weighting is used very often as it causes the importance of a neighbour to decay when the distance from the point of interest grows. Especially for thin boundary layer cells weighting is important

and inverse distance weighting already significantly increases the accuracy [19]. As noted before, not only the difference in distance from the centroid, but also the distribution around the centroid is of importance. Therefore, it is also possible to weight by the face area dotted with its normal as this is roughly inversely proportional to the degree of clustering in that direction [20]. In a more recent research by Syrakos et al. [20], they improved this version of weighting to counter the degree of clustering in a specific direction by the so-called direction weighting. For each specific neighbour, their normalized direction vector is dotted with all other normalized direction vectors of which the contributions are added to measure the alignment in that specific direction. If one direction is over represented, then the contribution of this neighbour is reduced accordingly. All these types of weighting increase the performance of the least squares method for many different grid types, but Syrakos et al. [20] found that there were inconsistencies. In their paper they give guidelines on which type of weighting to use and when to avoid certain types of weighting to prevent inconsistent behaviour of the weighted least squares method. Mavriplis [19] also mentioned possible inconsistencies with regards to weighted least squares schemes and explained that robustness issues can arise if the limitations of the method are not properly understood which is merely meant for further use in a finite volume method. In Figure (4), the left schematic denotes the node based least squares stencil which determines the gradient in the nodes. It should be mentioned that, similar to the Green-Gauss stencil shown on the right in Figure (3), the value $f(\vec{V})$, where $\vec{V}$ denotes the node coordinates of the node in question, is not known, whereas it is required for the computation of the least squares solution. Finally, it must be mentioned that, in contrast to the Green-Gauss method, the gradient can be evaluated at boundary cells with the least squares method, though be it a more one-sided approach with neighbours further away. Different approaches for gradient evaluation at boundary cells are examined by Wei et al. [21] and Chen et al. [22] which both mention an adjusted extended stencil for boundary cells or a ghost cell approach. All other approaches are mainly meant for the finite volume method, which presume known quantities on the boundaries or incorporate boundary conditions. The Green-Gauss method on the other hand, is not able to close a contour around the boundary point using other known grid points.

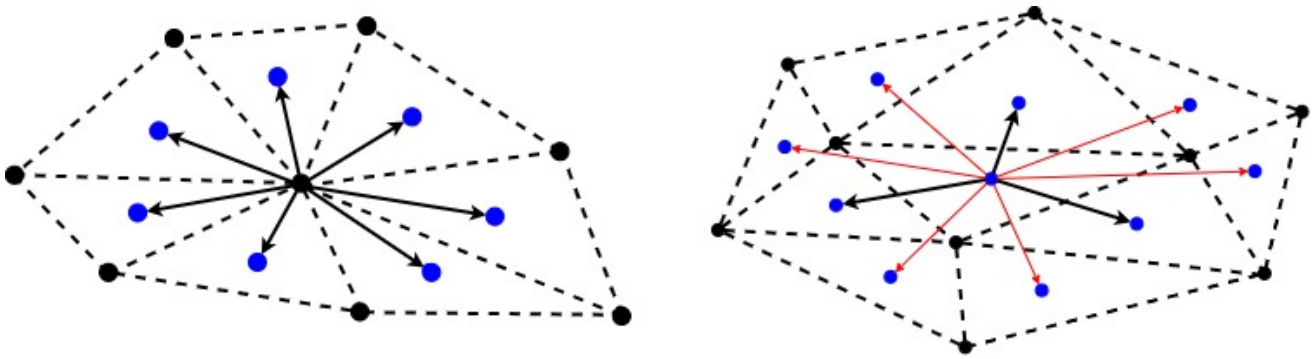


Figure 4: The Least Squares stencil for a node centered approach (left) and for a cell centered approach (right).

2.1.3 Other Surface Gradient Reconstruction Schemes

Apart from the two main branches in gradient reconstruction schemes, there are other methods that can also result in good reconstructions. Sozer et al. [12] described the so-called curvilinear gradient scheme of which the nearest neighbour stencil is mapped to a uniform Cartesian mesh and therefore imitates an equidistant orthogonal grid. This mapping is shown in Figure (5) where φ is the quantity of which the gradient is sought. A line between two different neighbouring cell centers forms a curvilinear direction ξ while another line, possibly starting from one of the same neighbouring cell centers, to another cell center forms the other curvilinear direction η. Using central differencing gives:

$$\frac{\partial \varphi}{\partial \xi} = \varphi_{i+1} - \varphi_{i-1}, \tag{15}$$

$$\frac{\partial \varphi}{\partial \eta} = \varphi_{j+1} - \varphi_{j-1}. \tag{16}$$

All possible stencil pairs i.e. the neighbouring cell centers that form the curvilinear directions, are chosen such that the determinant of the Jacobian matrix is as large as possible. These stencil pairs usually correspond with the curvilinear directions that are most orthogonal. The Jacobian is described as:

$$J = \begin{bmatrix} \frac{\partial x}{\partial \xi} & \frac{\partial y}{\partial \xi} \\ \frac{\partial x}{\partial \eta} & \frac{\partial y}{\partial \eta} \end{bmatrix} = \begin{bmatrix} (x_{i+1} - x_{i-1}) & (y_{i+1} - y_{i-1}) \\ (x_{j+1} - x_{j-1}) & (y_{j+1} - y_{j-1}) \end{bmatrix}. \tag{17}$$

Consequently, the gradient in Cartesian coordinates can be determined as:

$$\begin{Bmatrix} \frac{\partial \varphi}{\partial x} \\ \frac{\partial \varphi}{\partial y} \end{Bmatrix} = J^{-1} \begin{Bmatrix} \frac{\partial \varphi}{\partial \xi} \\ \frac{\partial \varphi}{\partial \eta} \end{Bmatrix} \tag{18}$$

Sozer et al. [12] tested the accuracy of the curvilinear gradient scheme, along with others, on many different cell types. On equilateral structured grids, the curvilinear scheme is significantly better than all other methods with an error two orders of magnitude lower. For all types of triangular grids and grids with very thin elements, the method achieves at least first order accuracy while performing in between the least squares method and the Green-Gauss method with respect to the error. Apart from the paper by Sozer et al. [12], no other papers are found in which this method is mentioned.

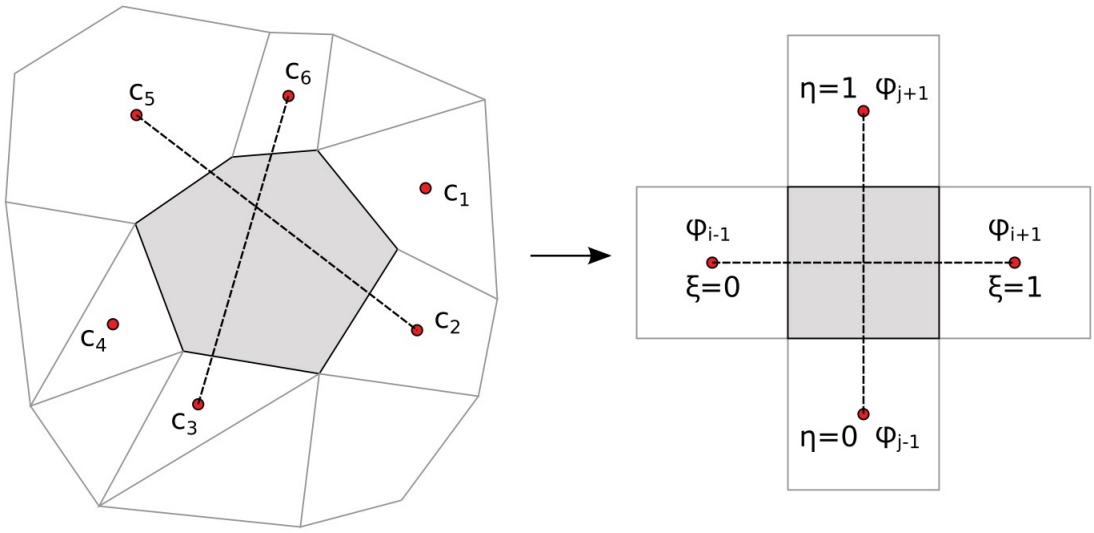


Figure 5: Curvilinear gradient stencil and mapping [12].

Another way to determine the gradient is by using regression based methods. Mancinelli et al. [23] described a linear regression method to determine the gradient at grid vertices using the function values at the vertex itself, v_i, and the surrounding vertices. With these values, for a given degree, a polynomial π_i is used to approximate the function in the vicinity of the vertex. In their work, quadratic polynomials are considered which can be denoted by:

$$\pi_i(x, y) = a_i x^2 + b_i y^2 + c_i xy + d_i x + e_i y + f_i. \tag{19}$$

This is a 2D representation in which the coefficients a_i, b_i, c_i, d_i, e_i and f_i are unknown. To solve for these unknowns, at each of the neighbouring vertices v_j, they write:

$$\pi_i(v_j) = f_j, \tag{20}$$

where v_i itself is also incorporated. This results in a linear system of equations which is treated as a weighted least squares problem, such that the importance of each vertex is incorporated. So, by incorporating $j-1$ neighbouring vertices and also including the vertex itself, the problem for vertex v_i can be written as $\vec{F} = [A]\vec{c_i}$ or:

$$\begin{Bmatrix} f_1 \\ f_2 \\ \vdots \\ f_j \end{Bmatrix} = \begin{bmatrix} x_1^2 & y_1^2 & x_1y_1 & x_1 & y_1 & 1 \\ x_2^2 & y_2^2 & x_2y_2 & x_2 & y_2 & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ x_j^2 & y_j^2 & x_jy_j & x_j & y_j & 1 \end{bmatrix} \begin{Bmatrix} a_i \\ b_i \\ \vdots \\ f_i \end{Bmatrix} \tag{21}$$

The coefficients in $\vec{c}$ which will result in the best fitting polynomial can be obtained using the exact same approach as in Section (2.1.2) resulting in:

$$\vec{c_i} = \left([A]^T[W][A]\right)^{-1}[A]^T[W]\vec{F}. \tag{22}$$

It should be noted that Mancinelli et al. [23] used the following weights:

$$w_i(j) = \frac{1}{L\sqrt{2\pi}} e^{-\frac{d_{ij}^2}{L^2}}, \tag{23}$$

in which d_{ij} denotes the distance from the vertex in question (v_i) to a neighbouring vertex v_j (which can also be the vertex itself). Moreover, L here denotes the average edge length in the mesh. This weighting is comparable to inverse distance weighting, but it is combined with a Gaussian decay [23]. Finally, the coefficients are known and therefore, the gradient of the function can be determined as:

$$\begin{Bmatrix} \frac{\partial \pi}{\partial x}(v_i) \\ \frac{\partial \pi}{\partial y}(v_i) \end{Bmatrix} = \begin{Bmatrix} 2a_ix + c_iy + d_i \\ 2b_iy + c_ix + e_i \end{Bmatrix}. \tag{24}$$

This approach is expandable to 3D which requires a larger system as the polynomial π_i now has additional terms and therefore unknowns. Additionally, with respect to the non-weighted least squares solution, the weighted least squares variant was found to be more accurate for all of their experiments. They tested this method with several other methods on structured, unstructured and anisotropic meshes for test functions consisting of sines and cosines in which the magnitude and frequency are controlled by parameters. The linear regression method appears to have a superior behaviour on anisotropic meshes and is overall a good method when evaluating interior points. However, the method does seem to suffer on the boundaries. Additionally, the method is computationally quite expensive and it requires function values in the nodes. Correa et al. [24] also discussed regression based methods. They used a similar approach as described above, but they also mention a 4D regression approach. This method seems to be able to determine the gradient at a vertex along with the unknown function value at that vertex. Weighting does not seem to be as effective in this case. In accordance with Mancinelli et al. [23], they also find that, apart from the 4D regression method, inverse distance weighted regression is accurate on highly irregular meshes, though it being a more expensive method.

2.2 Grid Analysis

The low-order panel method divides the surface in panels consisting of any number of nodes (triangular, quadrilateral etc.) and determines the values of the doublet singularity distribution on each panel by applying the boundary conditions in each of the collocation points just beneath the panels. For further

analysis, these doublet strength values are ascribed to the cell centers of each panel, while essentially this value is constant over the entire panel. All these panels together form a mesh of which the structure is important with regards to determining the surface gradient.

2.2.1 Grid Convergence Study

As discussed previously, the type of panel, in this case planar panels, influence the accuracy of the method. For curved surfaces, such as depicted in Figure (6), planar panels cannot exactly represent the actual surface and therefore, they can only approximate the surface. By decreasing the panel size, it is possible to fit much more panels and therefore, the approximation of the surface is more accurate as shown in Figure (6). This panelling of a surface is a type of discretisation and the difference between the actual surface and the approximate surface is therefore the discretisation error. Upon grid refinement i.e. the panels become smaller each refinement, this discretisation error should reach zero asymptotically [25].

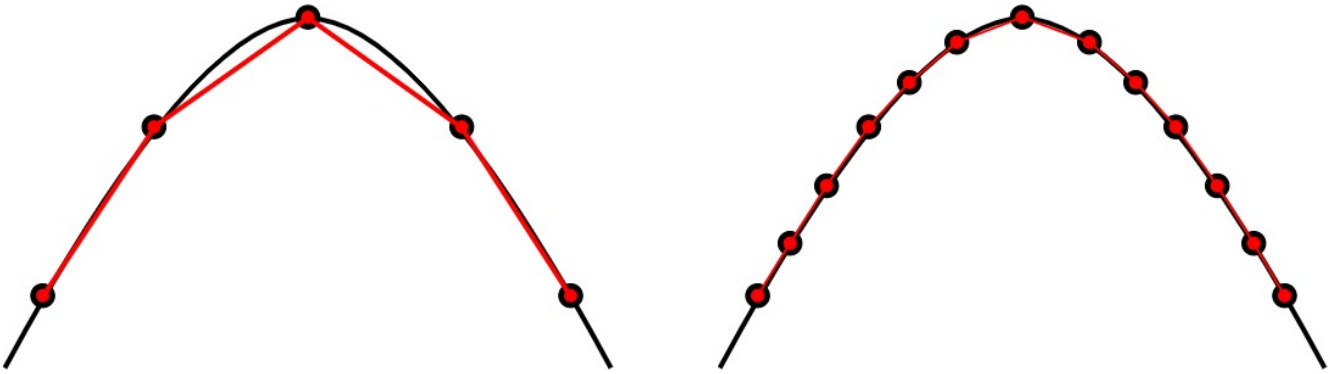


Figure 6: Approximation of a curved surface (black line) by planar panels (red lines and nodes) for big panels (left) and smaller panels (right).

Apart from approximating the surface, the panel midpoints or nodes can also be used to store information. Many methods that use neighbouring points for information, such as the methods discussed in Section (2.1), also benefit from grid refinement. For example, in Section (C) the doublet strengths are determined for each panel and these values are ascribed to the panel centers. Although for data storage these values are ascribed to the panel centers, these values can be seen as a constant distribution over the entire panel. This is also shown in Figure (7), where the blue rectangles essentially show the constant distribution on each panel. The red dots denote the panel centers where the actual values are stored and the lines connecting them approximate the black line, which represents the continuous version of the doublet strength distribution over the surface. Grid refinement will, in a similar fashion as approximating the curved surface in Figure (6), more accurately describe the real situation and eventually, the discretisation error (the error between the red line and the black line) should asymptotically reach zero. However, there are several problems when it comes to grid refinement. First of all, some methods, such as weighted least squares variants may result in inconsistent behaviour and do therefore not converge upon grid refinement [19], [20]. To check the consistency of the different formulations, the L_p-norm is used, which is described as:

$$L_p = \sqrt[p]{\frac{1}{N}\sum_{i=1}^{N}(f_{Ex,i} - f_{App,i})^p}, \tag{25}$$

for which $p = 1$, $p = 2$ and $p = \infty$ are popular choices. This norm determines the sum of the errors at every point, which can be done for decreasing grid sizes and the results can be plotted in a so-called log-log graph. The horizontal axis denotes the mesh size and the vertical axis denotes the L_p-norm. The log-log scale is beneficial as the behaviour of the L_p-norm can be observed for a decreasing mesh size. If

this graph reaches its asymptotic range, the approximation will converge to the exact solution and the slope of the graph can be computed to determine the corresponding order of accuracy. On the contrary, if the asymptotic range is never achieved, the graph will diverge resulting in an inconsistent approximation that never achieves the exact solution. Apart from convergence, this method also gives insight in the number of times a grid must be refined to achieve a certain error. For example, if an approximation is only required to reconstruct the solution to two decimals, then the corresponding amount of grid refinements can be found from the graph which minimises the computational costs.

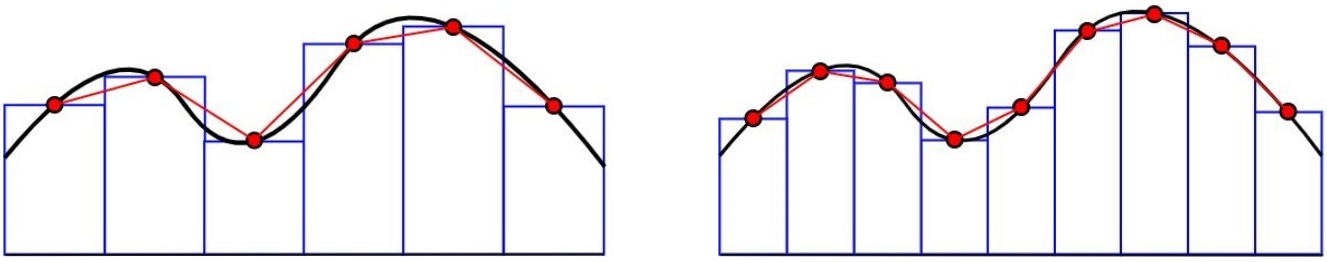


Figure 7: The function of a quantity (black line) over a surface approximated by constant strength panels (blue rectangles) of which the values are stored in the panel centers (red line) and how it is affected by grid refinement.

2.2.2 Structured, Unstructured, Primary and Secondary Grids

Although there are many different classifications when it comes to the structure of a grid, a distinction between a structured and an unstructured grid can always be made. In Figure (8), a small portion of an equidistant quadrilateral grid is depicted, where for each non-boundary node (i, j), it is possible to identify its four direct neighbouring nodes. This is known as the five point stencil which is often used to determine derivatives in the finite difference method. As long as the increments along the i- and j-direction remain constant, this stencil holds for all interior points. Essentially, as long as for every interior node it is possible to apply the same stencil and the nodes are numbered accordingly, a grid can be classified as structured. When looking at Figure (9), the directions i and j are now inclined. However, as long as the grid spacing in these directions remain constant over the entire grid, a similar five point stencil can be used. On these types of grids, due to the fact that there are neighbours of the same weight on either side of the point of interest, there are favourable truncation error cancellations as, for example happens in the finite difference method, increasing the order of accuracy.

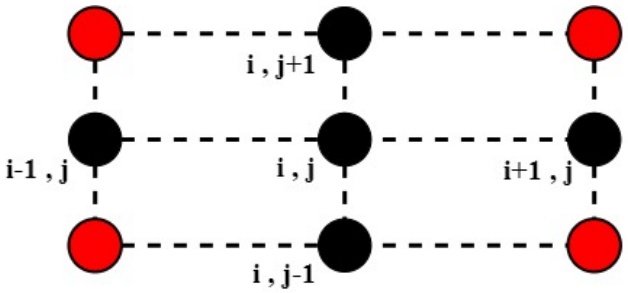


Figure 8: Depiction of the so-called five point stencil in a equidistant structured grid.

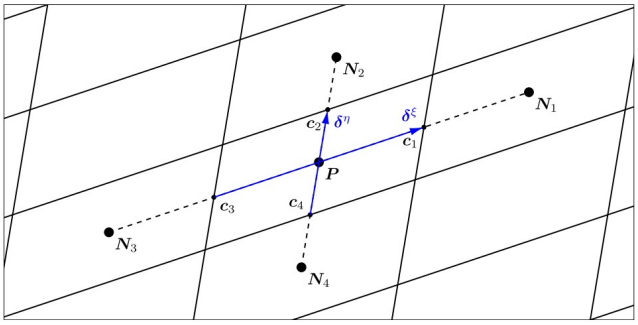


Figure 9: Depiction of a grid constructed with equidistant parallel grid lines [11].

Although the grid in Figure (9) is structured, the fact that the grid lines are not orthogonal suggests that the quality of the grid is degraded. Syrakos et al. [11] used three different grid quality indicators to determine the quality of a grid. In Figure (10), the blue arrow shows the face normal $\bar{n}_1$ of cell P. The centroid of face f_1 is denoted with $\vec{c}_1$ and $\vec{c_1}$ denotes the point on the line connecting the cell centroids $\vec{P}$

and $\vec{N}_1$ that is closest to the face centroid $\vec{c}_1$. So, in this case, face f_1 of cell P exhibits non-orthogonality if the line connecting the cell centroids $\vec{P}$ and $\vec{N}_1$ is not parallel to the face normal $\bar{n}_1$. The angle θ can be a measure of non-orthogonality in this case. Furthermore, the same face f_1 exhibits unevenness if the midpoint $\vec{m}_1$ of the line connecting the cell centroids $\vec{P}$ and $\vec{N}_1$ is not in the same position as $\vec{c_1'}$. A measure of unevenness can be defined as the distance between $\vec{m}_1$ and $\vec{c_1'}$ divided by the distance between the cell centroids $\vec{P}$ and $\vec{N}_1$. Finally, face f_1 is said to exhibit skewness if $\vec{c}_1$ and $\vec{c_1'}$ are not in the same position. This can be measured as the distance between $\vec{c}_1$ and $\vec{c_1'}$ divided by the distance between the cell centroids $\vec{P}$ and $\vec{N}_1$. These three indicators are not always affected individually as they are closely related [11]. The grid in Figure (9) is not skewed as $\vec{c}_1$ and $\vec{c_1'}$ coincide and it is also not uneven as $\vec{m}_1$ and $\vec{c_1'}$ coincide as well. Moreover, in Figure (11) a grid construction method is depicted that connects the intersections of two families of curves. These families of curves consist of smoothly distributed variables ξ and η which do not have to vary linearly in the domain and they also don't have to be constant along straight lines. These curves are constructed with equal intervals $\Delta\xi$ and $\Delta\eta$. Syrakos et al. [11] showed that, for these type of grid constructions, the cell geometry of a cell itself and of its neighbours locally approach the geometry of the grid shown in Figure (9) when the grid is refined. This is due to the fact that both skewness and unevenness tend to zero in the limit of $\Delta\xi$ and $\Delta\eta$ going to zero, as the grid lines become more and more parallel. From this, it can be deduced that, upon grid refinement, these types of grids will also be able to achieve higher order accuracy due to error cancellations. Note here, that this only holds for structured grids as both unevenness and skewness do not tend to zero for unstructured grids upon grid refinement. Although these grid quality indicators are not useful in combination with grid refinement on unstructured grids, they could help quantify how unstructured or irregular a grid actually is.

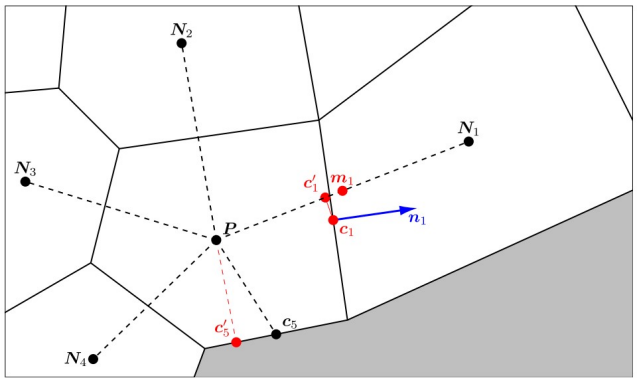


Figure 10: An unstructured grid with F neighbouring cells (N_i with i=1,2,...,F) around cell P from which the grid indicators are derived. The grey area represents a boundary [11].

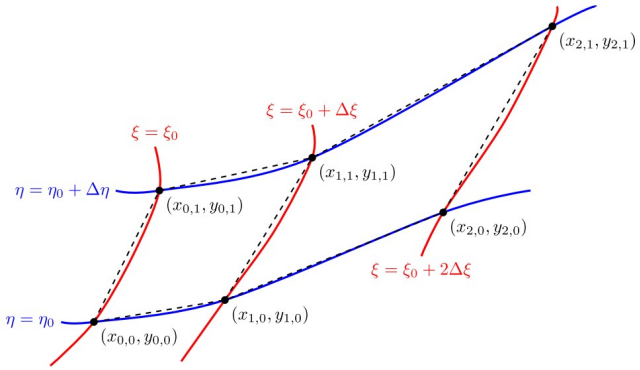
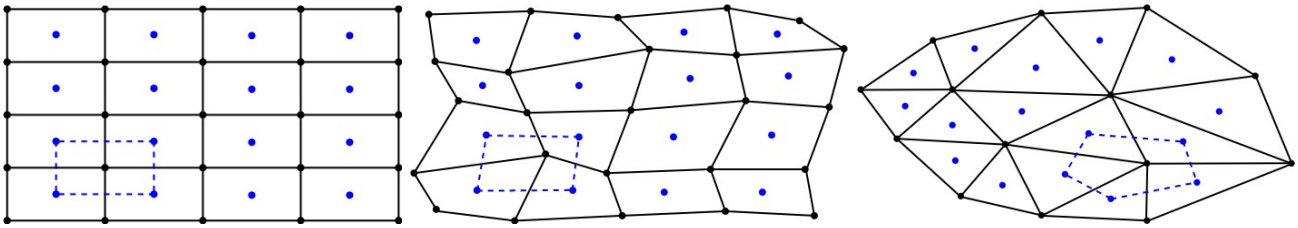


Figure 11: A grid (dashed black lines) constructed by smoothly distributed variables ξ and η. The values of ξ and η are constant along the curves (red and blue) with equal intervals $\Delta\xi$ and $\Delta\eta$ [11].

Similar to the gradient reconstruction schemes discussed in Section (2.1), grids can be observed as being cell centered (see Figure (10) with cell P) or as node centered e.g. a cell constructed around a node by neighbouring nodes or cell centers. Both can be observed in a single grid, where the primary grid is constructed from the nodes and the edges connecting them, while the so-called dual or secondary grid is constructed by using cell centers as 'nodes' and connecting them. In Figure (12), the primary grids are depicted in black, while the secondary grids are depicted in blue. First of all, note that in Subfigure (12.a.) the secondary grid is the same as the primary grid and that the node that now represents the cell center in the secondary grid is in the actual center. For both unstructured grids in Subfigures (12.b.) and (12.c.), the nodes that now represent the cell centers in the secondary grids are not in the centers of the cells. In contrast to the unstructured quadrilateral grid in Subfigure (12.b.), the unstructured triangular grid in Subfigure (12.c.) obtains completely different cells in its secondary grid with respect to its primary

grid. Furthermore, both of these grids are not structured. Despite the fact that the quadrilateral grid in Subfigure (12.b.) always has data points left, right, below and above a point of interest, the distances between these points are irregular as well as the fact that the points left, right, below and above are randomly shifted. Therefore, there is no stencil that can be used for all internal points and upon grid refinement, these effects will not disappear. Moreover, Subfigure (12.c.) is also not able to use a single stencil as the number of cells around a node are different per node.



12.a. Structured quadrilateral grid. **12.b.** Perturbed quadrilateral grid. **12.c.** Unstructured triangular grid.

Figure 12: Different grid types in which the main mesh consists of the black nodes and edges while the dual grid connects the blue cell centers with blue edges.

2.3 Data Structures

In Section (2.1), different reconstruction schemes for a gradient were discussed. These schemes require a data structure that is able to determine, for each node and cell center, which cell centers or nodes are direct neighbours or which neighbours are a bit further away. Furthermore, Section (2.2) shows that the structure of a grid can influence the stencil i.e. the ring of direct neighbouring nodes or cell centers, which can be different for each individual cell center or node. This means that the data structure should, for a given node or cell center, be able to navigate from this point to all of the surrounding neighbours efficiently.

2.3.1 General requirements and assumptions

As a start, some assumptions are made and some requirements are stated regarding the mesh. First of all, the mesh topology is fixed and must be proper i.e. the manner in which vertices are connected remains the same independent of changing vertex positions as shown in Figure (13). To ensure a proper topology, the neighbourhood of a vertex must be homeomorphic with a disk, which conforms with a 2-manifold [26], [27]. This means that, if a small sphere intersects with the mesh, a disk is formed by the intersection surface which is allowed to bend or stretch, but it may not tear as is shown in Figure (14).

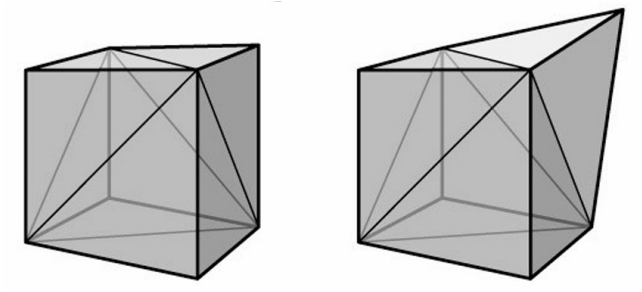


Figure 13: A fixed mesh topology for different vertex positions [26].

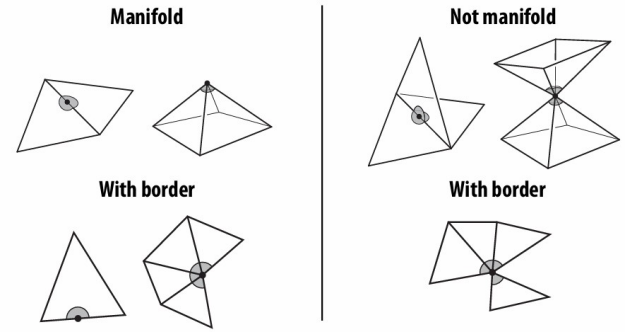


Figure 14: Examples of manifold and non-manifold meshes [26].

This has many implications, for example, every edge connects two vertices and separates two faces, while faces share only one vertex or two vertices at most. A ring of vertices, that are connected by edges, enclose a face while a ring of edges and faces surround a vertex. Additionally, the global topology is assumed to be proper which means that the mesh is closed and bounded. As a consequence, for meshes over closed surfaces such as a sphere, Euler's polyhedron formula holds:

$$F - E + V = 2, \tag{26}$$

in which F denotes the number of faces, E the number of edges and V the number of vertices. For example, the surface mesh over the closed box in Figure (13) has $F = 12$, $E = 18$ and $V = 8$ for which Equation (26) indeed results in two. Another implication of the 2-manifold, is that the surface is orientable i.e. if the boundary of each face is followed counterclockwise, every face only has one face normal. A non-orientable surface on the other hand, contains a Möbius strip on which a single face can have two opposing face normals making it impossible to distinguish between an inner and outer side of the surface. Finally, the required input data is assumed to be two arrays that contain the coordinates of the vertices, which lie on the actual surface, and an array that contains the nodes required to construct the faces.

2.3.2 Surface Grid Data Structures

Due to the increased complexity of mesh navigation on generally unstructured grids, an efficient data structure is essential. In 1972, Baumgart [28] introduced the winged-edge data structure that is shown in Figure (15).

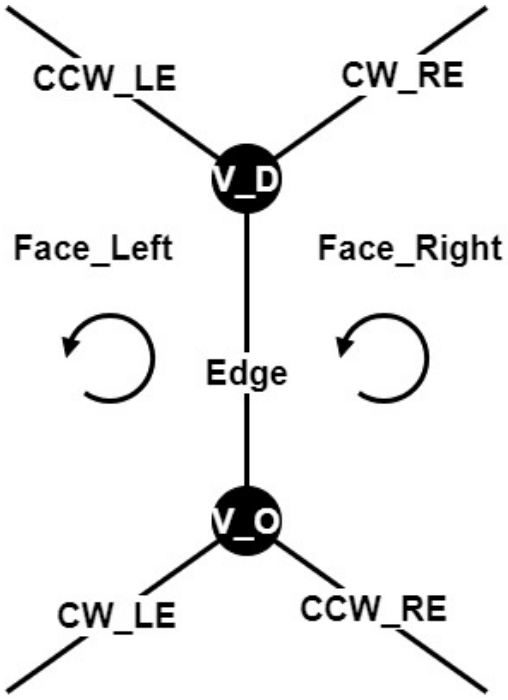


Figure 15: Schematic representation of the winged-edge data structure.

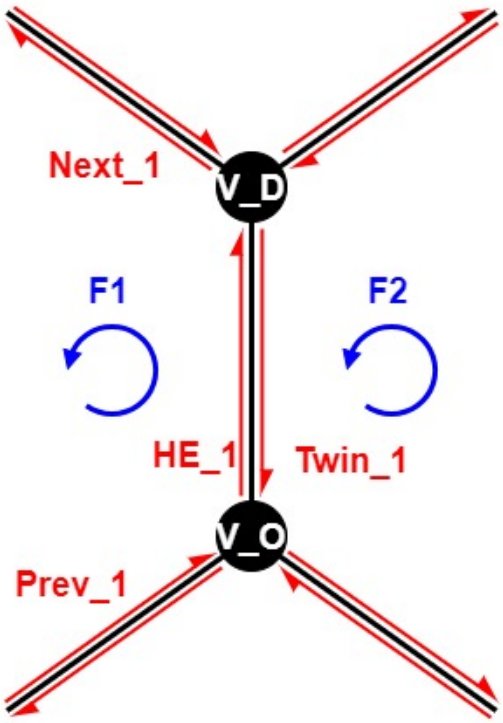
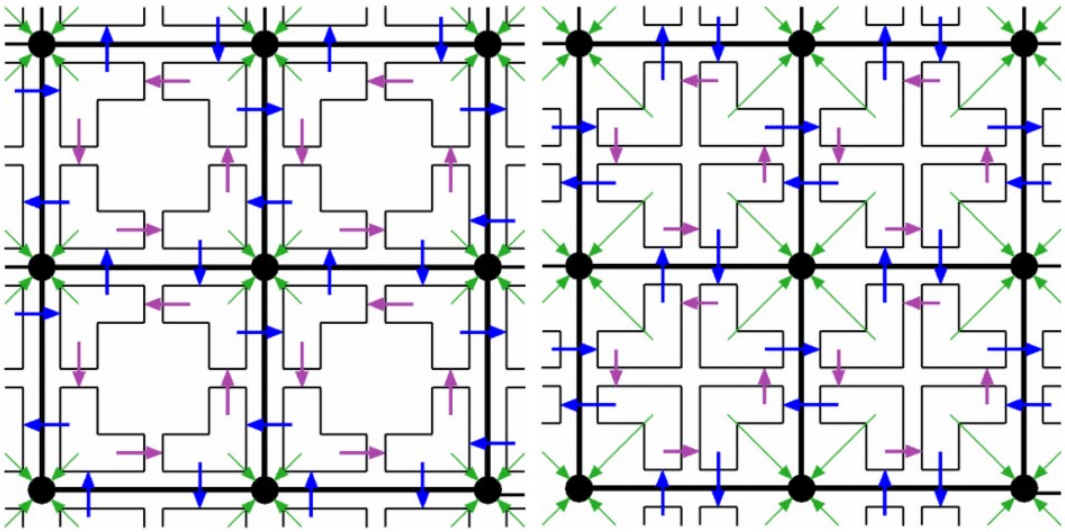


Figure 16: A schematic representation of the half-edge data structure.

In this method, each vertex stores its coordinates and an edge it is connected to. Additionally, each face stores a reference to one of its enclosing edges. Finally, every edge stores references to the vertices it connects (V_O and V_D), the faces it separates (Face_Left and Face_Right), the previous edges left and

right (CW_LE and CCW_RE respectively) and the next edges left and right (CCW_LE and CW_RE respectively). For a given face, one can iterate from that face to one of its enclosing edges. Consequently, from this edge, all other edges enclosing that face can be obtained and these edges can be used to determine the neighbouring faces. Similarly, one can start from a given vertex and determine to which vertices it is connected and which faces it is surrounded by. Another data structure, that is derived from the winged-edge data structure, is the half-edge data structure. This data structure is depicted in Figure (16), from which it can be seen that all edges are now split into two half-edges that point in opposite directions. Each of the vertices now store their coordinates and a reference to one of their outgoing half-edges (V_D may store the half-edge labeled Next_1), where boundary vertices must reference an outgoing half-edge that is part of the boundary. Similarly, every face stores a reference to one of its enclosing half-edges (F_1 may store HE_1). Lastly, every half-edge stores the vertex it points to, the face it encloses, the next half-edge in the face, its opposing half-edge and optionally the previous half-edge within its face (HE_1 may store V_D, F_1, Next_1 and Twin_1 respectively). Both of these data structures are able to perform queries, such as which faces surround this vertex, in constant time. This means that every step, so looking for the right index in an array, always takes the same amount of time independent of how big this array is. Therefore, the time this query takes is linear with the amount of faces surrounding this vertex [29]. Furthermore, both structures are able to perform local mesh operations such as mesh editing by inserting or deleting elements and flipping, splitting and collapsing edges. These operations are tools for mesh subdivision, mesh simplifications and mesh regularisation often used to obtain smooth grids in graphic design [26]. Between these two methods, the half-edge data structure is able to represent boundaries in a more sophisticated way with respect to the winged-edge data structure. Additionally, the half-edge representation is arguably easier to understand and is easier to initialise in contrast to the winged edge representation [27], [30]. Based on these data structures, more complex data structures are derived that take the secondary grid in consideration as well. For example, the quad-edge data structure that essentially consists of two half-edge data structures for both the primary and the secondary grid. When switching between the primary and the secondary grid, faces and vertices switch roles and edges can switch based on operators that consist of a 90 degree rotation, a flip, a twin and a counter clockwise rotation about its origin [26]. Another example is the split-edge data structure that is, again, very similar to the half-edge data structure, but instead of an edge split along its length, the edge is now split halfway.



17.a. Primary Grid **17.b.** Secondary Grid

Figure 17: Depiction of the corner data structure for both the primary and the secondary grid [27].

Finally, the corner data structure that splits the element into different corner blocks as depicted in Figure (17). Each corner block is aligned with one of the enclosing vertices of the element and reference

neighbouring corner blocks. The main benefit of the corner data structure, is that it works very similar for the secondary grid as only the roles of the vertices and the cell centers are reversed as well as the references by the blue and purple arrows.

3 Description of the Solution Methods and Test Problems

From the potential flow theory in combination with an (un)structured, low order panel method, a problem arises where the surface gradient must be determined in the nodes from a discrete set of doublet strengths stored at the face centers. In this section, the methods selected to solve the problem are discussed with the goal to obtain a gradient scheme that is able to give accurate approximations on both structured and unstructured grids.

3.1 The Half-Edge Data Structure

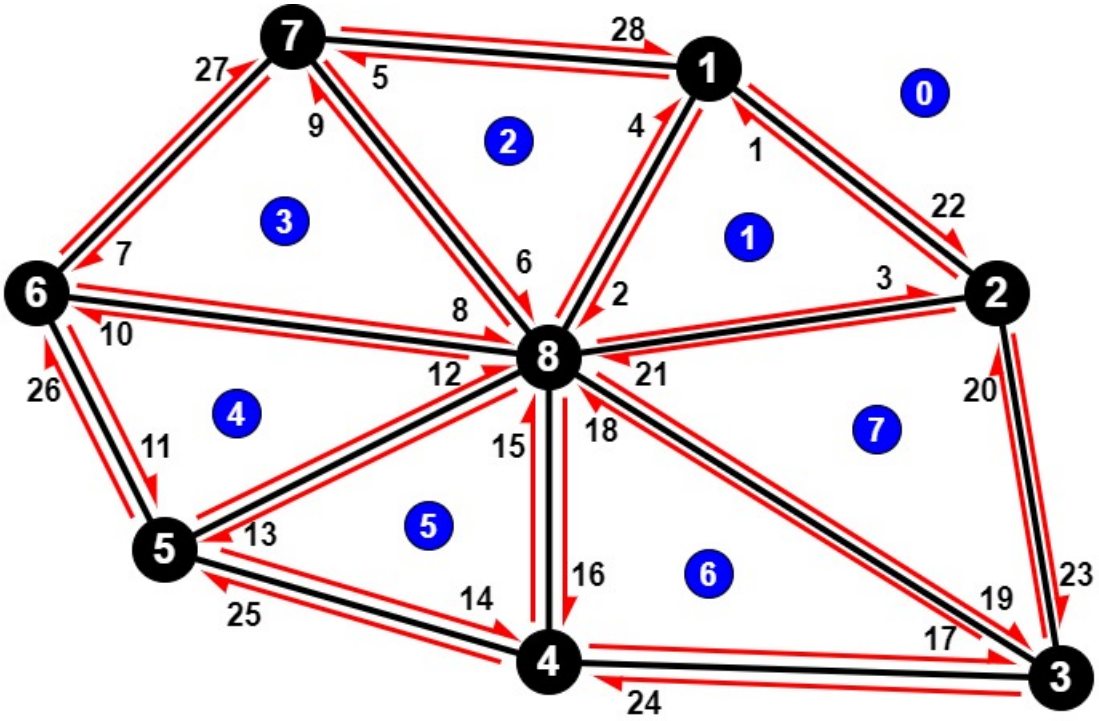


Figure 18: Half-edge data structure applied to an unstructured primary mesh with a boundary.

To navigate on both structured and unstructured meshes, a data structure is required that is able to adapt to any stencil. Several data structures are discussed and, apart from satisfying all requirements, the half-edge data structure is selected due to its ease of use and implementation, but also for how it is able to handle boundaries. Furthermore, it is able to determine all neighbouring cell centers which is already enough information to determine the secondary grid. The mesh given in Figure (18) is used to give a more in depth explanation of the half-edge data structure. First of all, the boundary can be seen as one big face i.e. face 0. In case the mesh is a plate with a hole in it, there is an outer boundary with face 0, while there is also an inner boundary with face -1 and so forth in case of more boundaries. As discussed previously, each face stores a reference to one of its enclosing half-edges as shown in Table (1). Similarly, each node stores its position by means of coordinates and it stores a reference to one of the half-edges that points away from this vertex. Table (2) denotes these references, where it can be seen that boundary nodes always reference a half-edge that lies on the boundary which is necessary to efficiently

classify boundaries [31]. Moreover, for all internal elements, the half-edges rotate counterclockwise, while for the outer boundary, the half-edges rotate clockwise. The corresponding references of some of the half-edges is denoted in Table (3). Essentially, these three arrays are enough to iterate over a whole mesh and can deal with any stencil. However, this data structure is easily extendable or editable. For example, to determine the surrounding faces of vertex eight in Figure (18), a so-called vertex circulator operation is used. As a start, vertex eight references the half-edge three which can already reference face one. As a follow up, half-edge three can reference its twin half-edge i.e. half-edge 21, which in turn can reference the next half-edge in the same face i.e. half-edge 19. Now half-edge 19 can reference its face and the same steps are repeated until the 'next' reference indicates half-edge three, meaning the cycle is closed. In this way, all surrounding faces are determined clockwise, which might not be preferable and therefore, instead of storing references to the next half-edge in each face, references to the previous half-edge in each face can be stored. Moreover, it is also possible to determine the vertices that are connected to vertex eight, or which faces are direct neighbours of face five and so on.

Table 1: Face array.

Face_ID	HE_Ref
0	22
1	1
2	4
3	7
4	10
5	13
6	16
7	19

Table 2: Vertex array.

Vertex_ID	HE_Ref
1	22
2	23
3	24
4	25
5	26
6	27
7	28
8	3

Table 3: The first few rows of the half-edge array.

HE_ID	Vertex_Ref	Face_Ref	Next_Ref	Twin_Ref
1	1	1	2	22
2	8	1	3	4
3	2	1	1	21
4	1	2	5	2
5	7	2	6	28
6	8	2	4	9
7	6	3	8	27
8	8	3	9	10

3.2 The Green-Gauss Surface Gradient Method

The gradient theorem, as given in Equation (2), is described for a volumetric element, while it is also possible to apply it to surface elements. To determine the surface gradient of the dipole strengths, van Garrel [4] used the gradient theorem on surface elements in which Equation (2) is reduced to:

$$\iint_{S_C} \vec{\nabla}_S \mu \, dS = \int_{L_C} \mu \bar{\nu} \, dl. \tag{27}$$

As the surface gradient must be determined in the nodes, it is important to understand that the stencil that is depicted on the left in Figure (3) will be used. In the equation above, $\bar{\nu}$ denotes the outward unit normal at each point on L. This can be determined from:

$$\bar{\nu} = \frac{\vec{\tau} \times \bar{n}}{|\vec{\tau} \times \bar{n}|}, \tag{28}$$

of which the definitions are given in Figure (2). Using this definition, the vertex unit normal $\bar{n}$ is leading as it represents the unit normal to the actual surface. Equation (27) can be rewritten, as a cell has $k = 1, 2, ..., F$ sides with a constant normal $\bar{\nu}$, so:

$$\iint_{S_C} \vec{\nabla}_S \mu \, dS = \sum_{k=1}^{F} \left(\bar{\nu}_k \int_{L_k} \mu \, dl \right). \tag{29}$$

Furthermore, the midpoint integration rule can be used to state:

$$\frac{1}{S_C}\iint_{S_C}\vec{\nabla}_s\mu\,dS = \vec{\nabla}_s\mu(\vec{C}) + \vec{\mathcal{O}}(h^2) \tag{30}$$

and

$$\frac{1}{L_k}\int_{L_k}\mu\,dl = \mu(\vec{c}_k) + \mathcal{O}(h^2). \tag{31}$$

In these equations h is a measure of the characteristic mesh size [11]. Similar as before, $\vec{C}$ denotes the centroid of the cell and $\vec{c}_k$ denotes the centroid of each of the edges. Using that $S_C \sim \mathcal{O}(h^2)$ and $L_k \sim \mathcal{O}(h)$, these equations can be written as:

$$\iint_{S_C}\vec{\nabla}_s\mu\,dS = \vec{\nabla}_s\mu(\vec{C})S_C + \vec{\mathcal{O}}(h^4) \tag{32}$$

and

$$\int_{L_k}\mu\,dl = \mu(\vec{c}_k)L_k + \mathcal{O}(h^3) \tag{33}$$

respectively. Substitution of these equation in Equation (29) gives:

$$\vec{\nabla}_s\mu(\vec{C})S_C + \vec{\mathcal{O}}(h^4) = \sum_{k=1}^{F}\bar{\nu}_k\mu(\vec{c}_k)L_k + \vec{\mathcal{O}}(h^3). \tag{34}$$

By dividing both sides of the equation with S_C, the Green-Gauss gradient reconstruction scheme is obtained:

$$\vec{\nabla}_s\mu(\vec{C}) = \frac{1}{S_C}\sum_{k=1}^{F}\bar{\nu}_k\mu(\vec{c}_k)L_k + \vec{\mathcal{O}}(h). \tag{35}$$

So, first of all, to determine the surface gradient, several quantities must be determined. As a start, the area of each panel is required. To determine the unit outward normal on the boundaries of the element ($\bar{\nu}_k$), the vertex normal ($\bar{n}$) is required as well as the tangential vector ($\vec{\tau}$). Additionally, the function value μ must now be known at the center of each of the edges i.e. the middle of each pair of cell centers. Finally, also the length of each edge must be known, which is essentially the length of $\vec{\tau}$. Second of all, the $\vec{\mathcal{O}}(h)$ term can be dropped leaving a first order approximation. As a final remark, h denotes a characteristic mesh size, which is straightforward for a structured grid, but less simple when it comes to arbitrary unstructured grids as not all mesh elements have the same size.

3.2.1 Required Quantities for the Green-Gauss Method

As can be seen from Figure (2), the normal necessary to compute the surface gradient with the Green-Gauss method, must be known at the vertices of the grid. For flat (planar) grids, these normals are the same as for the panel normals, but for curved grids this does not hold. A solution is to use the normals of the neighbouring panels in some sort of (weighted) average. So, first of all, the normals for each panel can be determined, which also directly influence the half-edge data structure to rotate clockwise or counterclockwise. When looking at Figure (19), two types of right hand rules are used. First of all, when pointing the index finger in the direction of $\vec{a}$ and the middle finger in the direction of $\vec{b}$, the cross product $\vec{a}\times\vec{b}$ points in the direction of thumbs up i.e. the normal is perpendicular to both vectors $\vec{a}$ and $\vec{b}$. As the direction is the same on any point of the panel, this vector is placed at the panel center and to create a unit vector it is divided by its own length. Pointing the thumb of the right hand in the direction

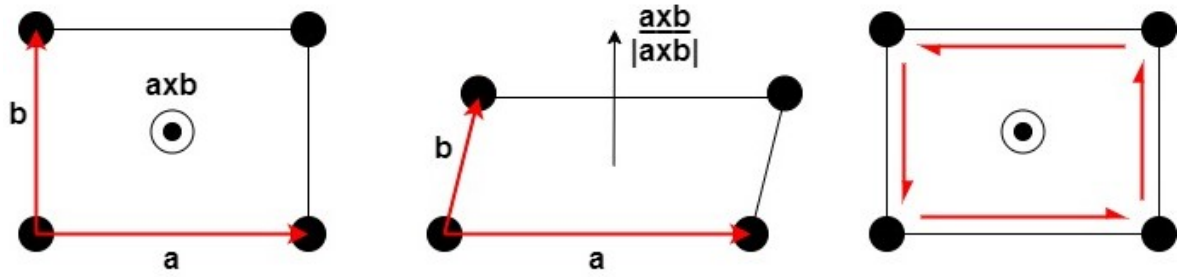


Figure 19: Top view of the cross product (left), side view of the resulting panel normal (middle) and the top view of the counterclockwise half-edges (right).

of this resulting unit normal, the fingers can only rotate around the thumb in counterclockwise direction which is therefore the orientation used in the half-edge data structure.
Some additional panel quantities that are required are the panel centers and the panel areas. The panel center coordinates are simply determined as the average of all the panel node coordinates. So, for a panel with $k = 1, 2..., K$ nodes ($\vec{N}_k$), its center $\vec{C}$ is determined as:

$$\vec{C} = \frac{1}{K} \sum_{k=1}^{K} \vec{N}_k. \tag{36}$$

To determine the area, each element is divided into triangles by connecting the panel center to the panel nodes with vectors r_k as shown in Figure (20). As the magnitude of the cross product between two vectors results in the area of a parallelogram, half of it will result in the area of the triangle. So, by summing the areas of the triangles, the area of the panel is obtained which can be denoted as:

$$A_{Panel} = \sum_{k=1}^{K} \left| \frac{(\vec{r}_k - \vec{r}_C) \times (\vec{r}_{k+1} - \vec{r}_C)}{2} \right|, \tag{37}$$

where $\vec{r}_C$ denotes the cell center vector, $\vec{r}_k$ denotes the vector from the panel center to node k and $\vec{r}_{k+1}$ to node $k + 1$ for which it should be noted that $\vec{r}_{k+1} = \vec{r}_1$ if $k = K$ [12].

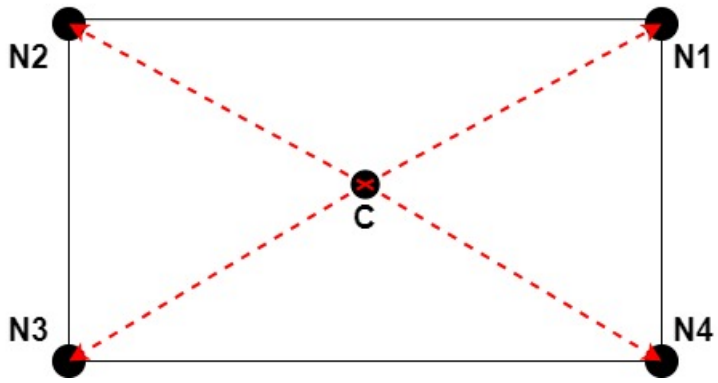


Figure 20: Depiction of the triangularisation of the panels using the panel centers.

Now, as these quantities are known, the vertex normal can be determined. So, when looking at Figure (2), it can be reasoned that the average of the surrounding panel normals will result in a good approximation for the vertex normal. However, for grids that are made up of different elements, possibly with big size differences, the average of the panel normals can result in worse approximations [32]. As a result, the vertex normals are determined as a weighted average of the panel normals as:

$$\vec{n}_v = \frac{\sum_{k=1}^{K} \bar{n}_k w_k}{\sum_{k=1}^{K} w_k} \quad \text{and} \quad \bar{n}_v = \frac{\vec{n}_k}{|\vec{n}_v|}. \tag{38}$$

In this equation, there are $k = 1, 2, ..., K$ surrounding faces of which the normal ($\bar{n}_k$) must be taken into account. Additionally, w_k denotes the weighting factor, $\vec{n}_k$ denotes a non-unit vertex normal and $\bar{n}_v$ denotes the unit vertex normal. There are several different weight factors that can be used, but it is chosen to base the weight factors on the area of the panels i.e.

$$w_k = A_k, \tag{39}$$

which deems larger panels to be more important. When looking at the exaggerated example in Figure (21), it can be seen that for big panel size differences, a panel normal averaged vertex normal results in fairly even curvature distributions, while for a vertex normal reconstructed by area weighted panel normals most of the curvature is ascribed to the smaller panel which is arguably more natural.

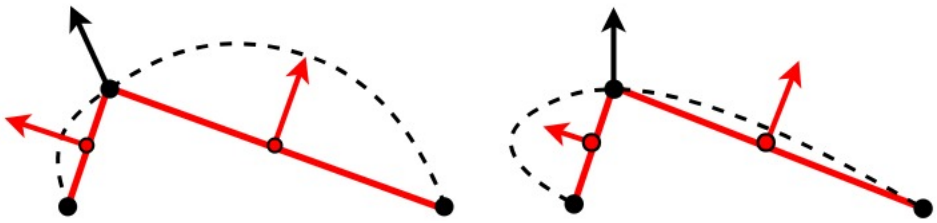


Figure 21: Exaggerated depiction of the surface reconstruction (striped black line) by averaging panel normal vectors (left) and by panel area weighted normal vectors (right). The black arrow depicts the vertex normal, while the red arrows depict the panel normals.

Finally, a measure for the characteristic mesh size h must be determined. Here, it is chosen to use:

$$h = \sqrt{\frac{1}{F} \sum_{f=1}^{F} A_f}, \tag{40}$$

which divides the summed area of all panels by the number of panels F of which the square root is taken to obtain a characteristic length.

3.3 Least Squares Method

Apart from applying the Green-Gauss method to reconstruct the gradient, also a least squares method is applied. When looking at Figure (4), the stencil depicted on the left will be considered. Equation (14) will be used:

$$\vec{\nabla}\mu(\vec{V}) = [G]^{-1}[A]^T \vec{b} \quad \text{with} \quad [G] = [A]^T[A]. \tag{41}$$

The vector $\vec{b}$ here denotes the difference between the function value at each of the neighbouring cell centers ($\vec{N}_k$) and the function value at the vertex $\vec{V}$. Using the panel method, the function value at the vertices is not known and should therefore be approximated. However, in this research, the function values are assumed to be known at the vertices. Moreover, all other properties stored in the matrices $[G]$ and $[A]$ are known. Note, however, that this is the surface gradient with a spurious component in the normal direction which must be removed [33]. Therefore, the following transformation can be applied:

$$\vec{\nabla}_s\mu(\vec{V}) = \vec{\nabla}\mu(\vec{V}) - \bar{n}_v \left(\bar{n}_v \cdot \vec{\nabla}\mu(\vec{V})\right), \tag{42}$$

which essentially subtracts the component normal to the surface [34].

3.4 Test Cases and Mesh Generation

Both methods, discussed in Sections (3.2) and (3.3), are able to approximate the (surface) gradient on both structured and unstructured grids. To test their behaviour and accuracy, different tests are performed on meshes with decreasing mesh size. To create different meshes, the open source, automatic mesh generation program GMSH [35] is used. For quadrilateral surface meshes the 'Quasi-Structured Quad' algorithm is used, while for triangular surface meshes the 'Delaunay' algorithm is used. To obtain refined meshes, the option 'refine by splitting' is used which approximately quadruples the number of elements in the mesh. First of all, a test is performed to validate that the sum of all the panel areas converges to the actual surface area upon grid refinement. This can help validating if the half-edge data structure is implemented correctly. Moreover, a flat equidistant quadrilateral grid is used as a benchmark test as all available data around a vertex is equally spaced, equally distanced and the vertex is centered with regards to the secondary grid as depicted in Figure (22). Additionally, curvature effects are avoided by considering the mesh to be flat which results in an optimal mesh that suits as a reference to less optimal meshes. Secondly, a grid constructed with triangles is used as a general unstructured grid which is shown in Figure (23). For vertices in this grid, all surrounding neighbouring cell centers do not necessarily have to be equally spaced or distanced and the vertex does not have to be centered with regards to the secondary grid. On this type of grid, favourable error cancellations, as discussed in Section (2.2.2), will not happen, but these grid irregularities are not that pronounced as most cells in Figure (23) are nicely formed. As learned from the studies by Syrakos et al. [11], [20], perturbation of the nodes of existing meshes will significant degrade the quality of the grids. Therefore, the nodes of the equidistant quadrilateral grid and the general triangular grid of Figures (22) and (23) are displaced as is depicted in Figures (24) and (25) respectively. To each of the node coordinates, a small value is added that lies in the range of $[-0.25h\,, 0.25h]$ and $[-0.2h\,, 0.2h]$, for quadrilaterals and triangles respectively. Here, h denotes the characteristic mesh size and these ranges are defined in such a way that the elements remain convex polygonals [11]. The degradation of the grid quality is especially observable when looking at the difference of both quadrilateral grids. All favourable properties of the equidistant quadrilateral mesh disappear due to the effect of the node perturbations and even with grid refinement these effects do not diminish. However, the main effect caused by these perturbations, is the displacement of the vertex from the actual cell centers of the secondary grid to a point, in most cases, not even close to the cell centers of the secondary grid.

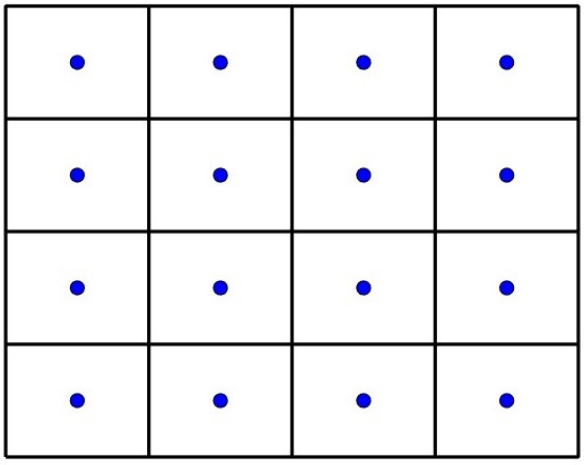


Figure 22: Equidistant quadrilateral grid.

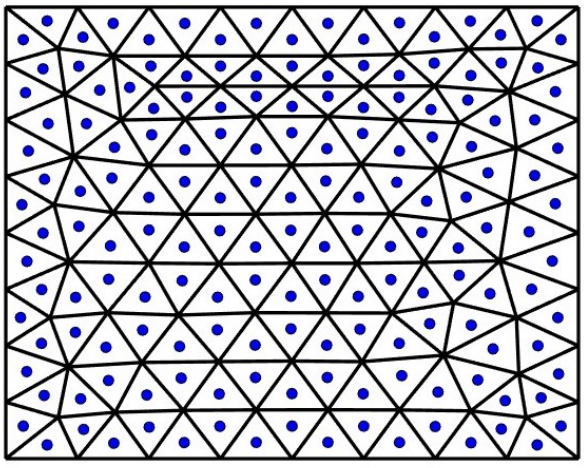


Figure 23: General triangular grid.

Similarly, perturbation of the nodes of the general triangular grid degrades the grid quality. The cells are much more deformed and the main consequence lies in the fact that the vertices that form the cell

centers of the secondary grid are, for most cases, not centered at all. On these grids, the effects of badly formed cells and non-centered cell centers can be evaluated without the effects of curvature.

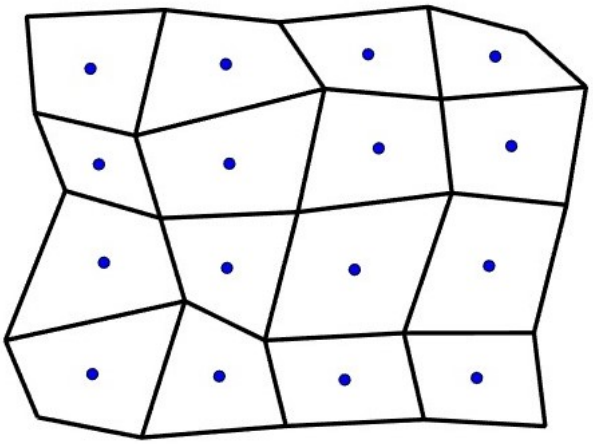


Figure 24: Perturbed variant of the equidistant quadrilateral grid.

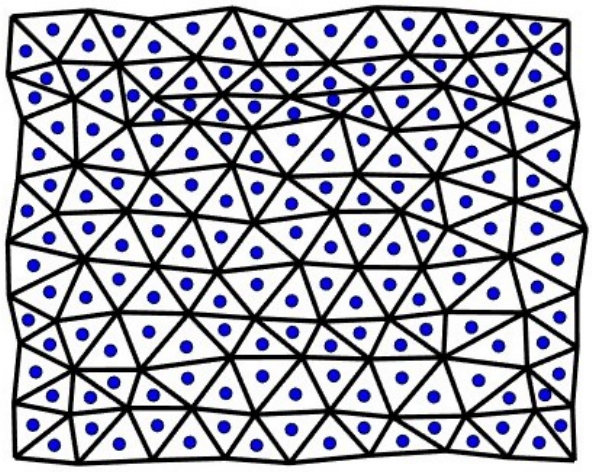


Figure 25: Perturbed variant of the general triangular grid.

For non-curved surfaces, the terms in Equation (35) are all exact as the mesh over the surface is exactly the same as the actual surface. However, in reality most surfaces are curved and therefore, the resulting grids are also curved. As shown in Figure (6), non-curved panels can only approximate a curved surface and cannot exactly represent the actual surface. This means that the terms in Equation (35) are not exact as the panels are not an exact representation of the surface. Only upon grid refinement, the accuracy of the approximation increases as the panels are more aligned with the actual surface and therefore also the terms in Equation (35) increase in accuracy, but remain an approximation. As a start, the z-coordinates of an equidistant quadrilateral grid, as shown in Figure (22), are determined as $z = 3x^2$. The resulting mesh is shown in Figure (26), from which it can be seen that the curvature is restricted to the x- and z-direction while the y-direction is independent of the curvature.

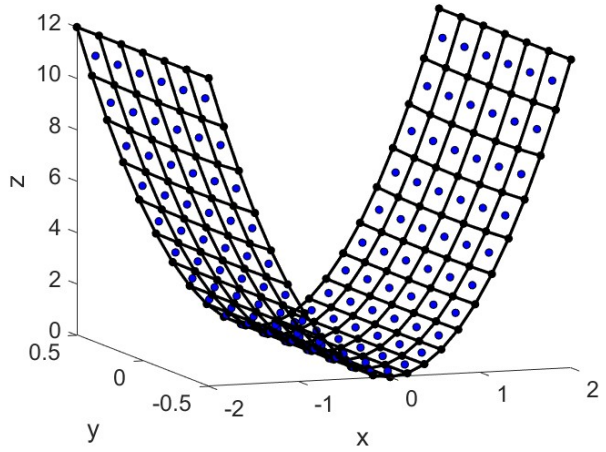


Figure 26: Equidistant quadrilateral mesh over a curved surface.

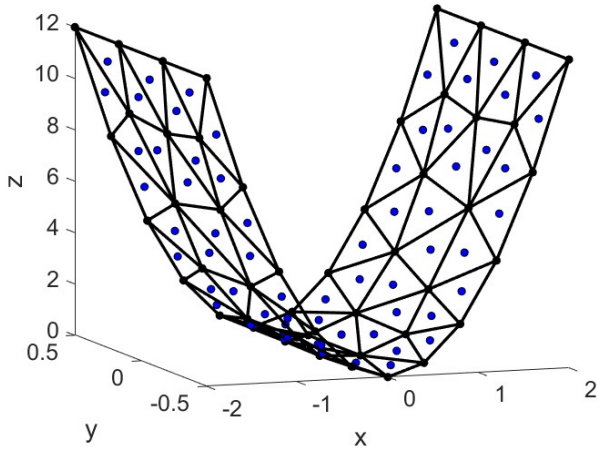


Figure 27: General triangular mesh over a curved surface.

Similarly, a general triangular grid as shown in Figure (23) is curved by determining the z-coordinates as $z = 3x^2$ which results in the grid shown in Figure (27). Furthermore, to test the effect of curvature

in all directions, a quadrilateral and a triangular mesh over the surface of a sphere are considered which are shown in Figures (28) and (29) respectively.

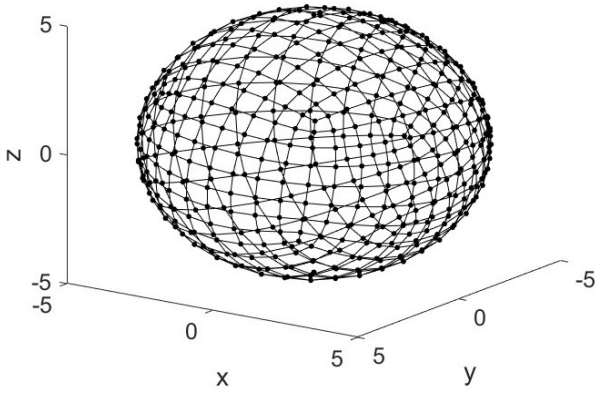


Figure 28: Quadrilateral mesh over a sphere.

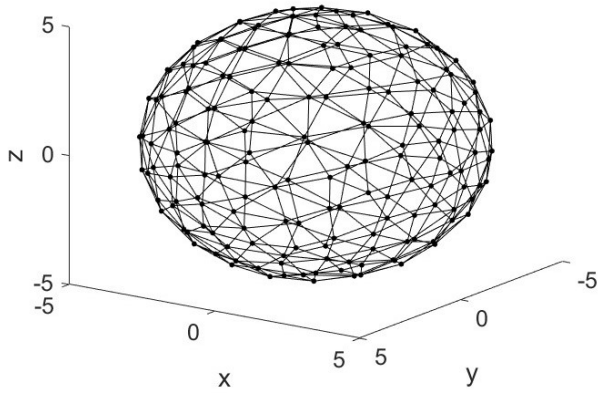


Figure 29: Triangular mesh over a sphere.

3.4.1 Test Functions

Apart from testing different grids to determine the behaviour of the surface gradient reconstruction schemes, also the function values stored at the cell centers will be determined from different functions. In this way, it is possible to determine whether either of the methods is able to reproduce the surface gradient to a linear, quadratic or higher order function. Based on the results of Sozer et al. [12] and Syrakos et al. [11], [20], it is chosen to use arbitrary linear, bi-linear , quadratic, cubic and sine/cosine functions which are given by

$$f_l(x,y,z) = 2x + 3y - 4z + 1, \tag{43}$$

$$f_b(x,y,z) = 2x + 4xy + 3y - 2xz + 5yz - 3z + 1, \tag{44}$$

$$f_q(x,y,z) = 2x^2 - 3y^2 + 4z^2 + 4, \tag{45}$$

$$f_c(x,y,z) = x^3 - 2y^3 + 6z^3 - 9, \tag{46}$$

$$f_s(x,y,z) = 5\sin(2x) + 2\cos(3y) - 2\sin(z) + 5, \tag{47}$$

respectively. According to Syrakos et al. [11], on flat equidistant quadrilateral grids as depicted in Figure (22), the surface gradient of linear, bi-linear and quadratic functions should be reconstructed exactly by the Green-Gauss surface gradient scheme and the least squares surface gradient scheme. As a consequence of using the midpoint integration rule in the Green-Gauss method and truncating the Taylor series in the least squares method, both schemes should not be able to exactly reconstruct the surface gradient for cubic or higher order functions. However, for these functions it is essential that upon grid refinement the error between the approximated surface gradient and the exact surface gradient diminishes.

4 Results and Discussion

4.1 Validation and Benchmark Tests

As mentioned previously, the first test is used to validate that the the sum of the panel areas converges to the actual area of the surface upon grid refinement. For the meshes depicted in Figures (28) and (29), the grid refinement is applied and the corresponding result is plotted in Figure (30), where h denotes the characteristic length as described in Equation (40) and $L_{ref} = 10$ (the diameter of the sphere) is used for non-dimensionalisation. The vertical axis denotes the error which is determined with the L_2 norm as described in Equation (25), which is divided by a reference area (the surface area of the sphere i.e. $A_{ref} = 314.15$) to make it dimensionless.

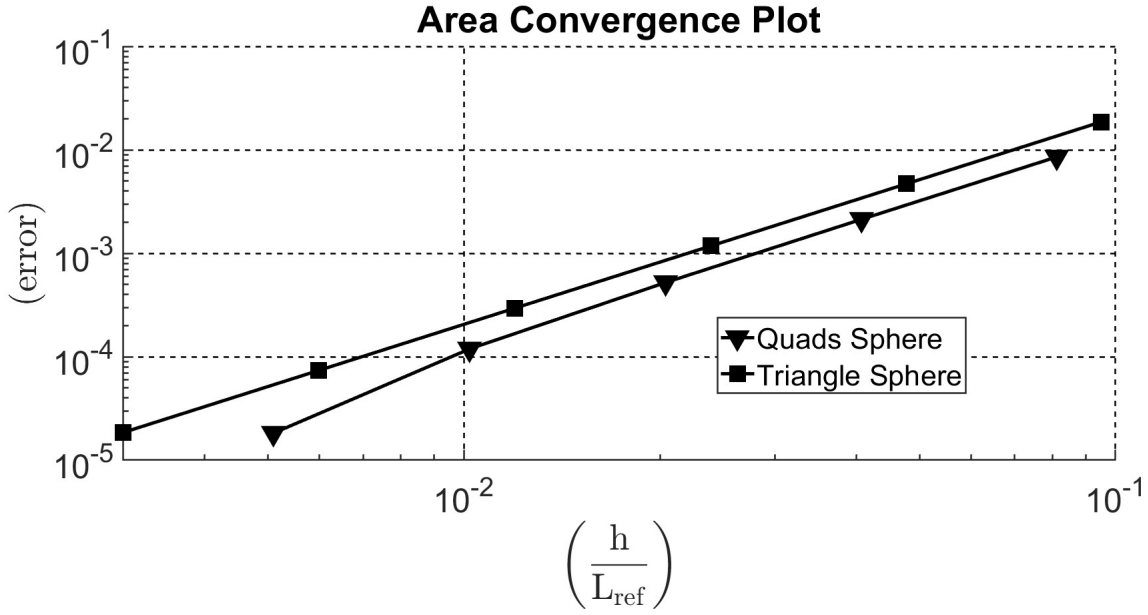


Figure 30: Convergence of the panel surface area approximation with respect to the real surface as a function of the non-dimensional characteristic length.

From Figure (30), it can be deduced that for both quadrilateral and triangular panels, the sum of all panel areas will converge to to real surface area. Both errors are $\mathcal{O}(h^2)$ which is to be expected for approximating a quadratic surface by flat panels. The quadrilateral mesh has a slightly smaller error, which seems in accordance with Figures (28) and (29) as the quadrilateral mesh over the sphere is initially already visibly smoother than the triangular mesh. As a benchmark test, both the Green-Gauss method and the unweighted least squares method, are used on a flat equidistant quadrilateral mesh for different functions and the results are depicted in Figures (31) and (32) respectively. It should be noted that for flat meshes, the z-components of the functions are neglected and that $L_{ref} = 1$. Both surface gradient reconstruction methods are able to exactly reconstruct the surface gradients of the linear, bi-linear and the quadratic functions on this mesh. These results are expected due to the quality of the grid. In addition, the cubic and sine/cosine functions are not exactly representable, but the errors are of $\mathcal{O}(h^2)$. Although not completely equal, both graphs are remarkably similar which was also observed by Sozer et al. [12] and Syrakos et al. [11]. Furthermore, it can be seen from these figures that both, the cubic and the sine/cosine functions cause lower errors in the x-direction with regards to the y-direction. When looking at these functions, it can be seen that the magnitude is more pronounced for the cubic function while the frequency is more pronounced for the sine/cosine function. These results were also obtained by Mancinelli et al. [23] who tested functions with varying amplitudes and frequencies.

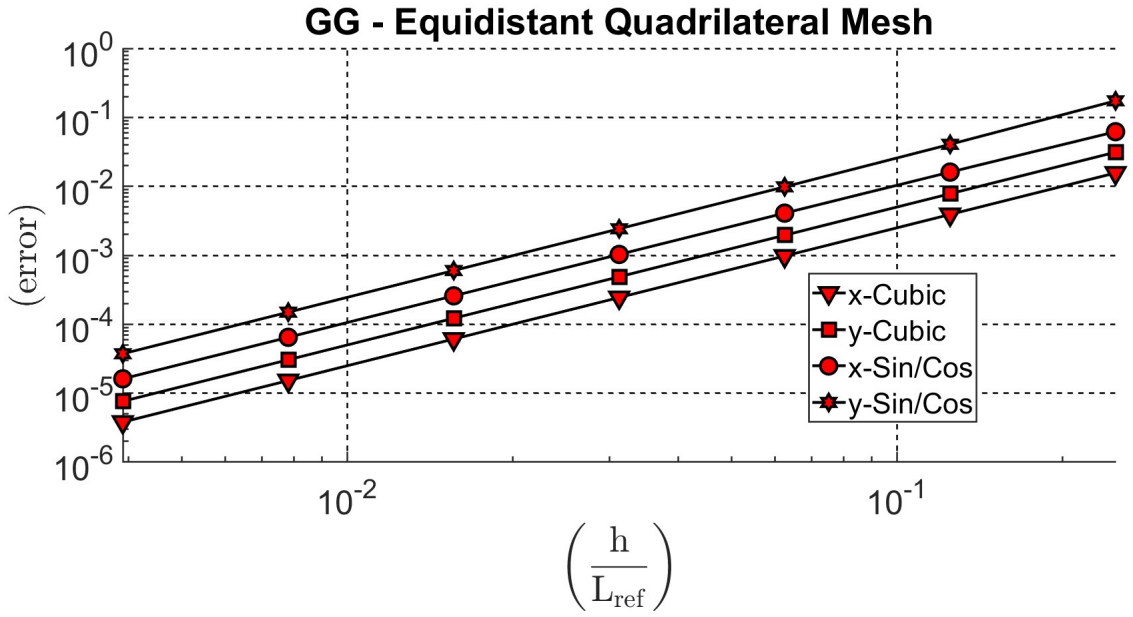


Figure 31: Surface gradient approximation error for the Green-Gauss method on an equidistant quadrilateral mesh as a function of the non-dimensional characteristic length.

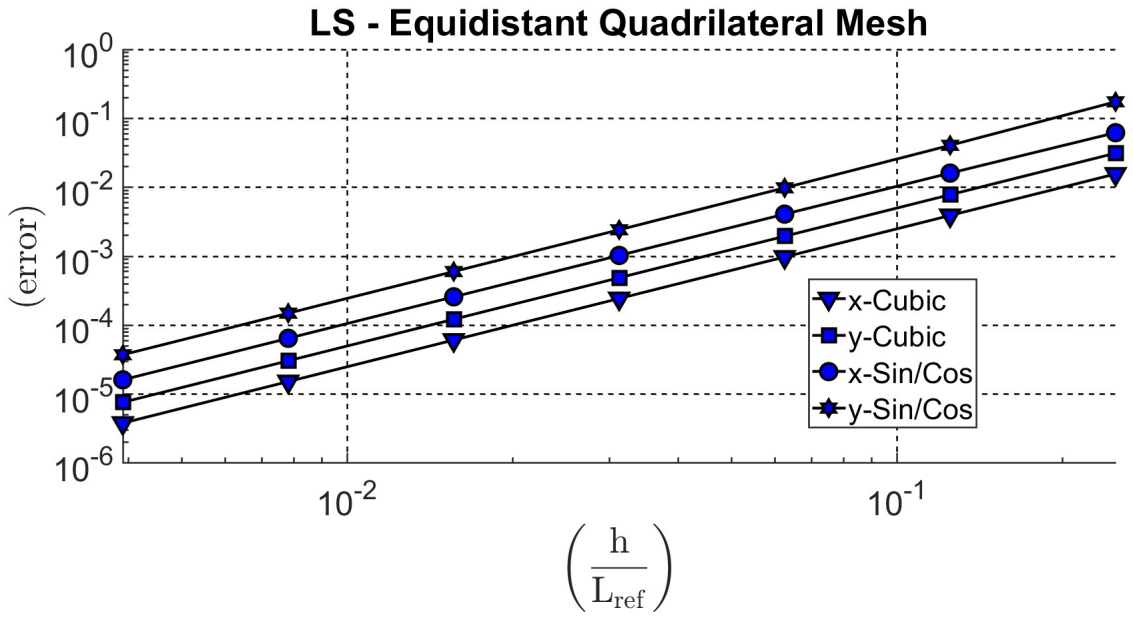


Figure 32: Surface gradient approximation error for the least squares method on an equidistant quadrilateral mesh as a function of the non-dimensional characteristic length.

4.2 Flat Mesh Test Cases

The function values of the functions given in Section (3.4.1) are also determined on the general unstructured triangular mesh as shown in Figure (23). To determine the surface gradient, the Green-Gauss surface gradient reconstruction scheme is applied and the unweighted least squares gradient reconstruction scheme of which the results, regarding the x-direction, are plotted in Figures (33) and (34). The results regarding the y-direction can be found in Appendix (E) as these result are similar to the results regarding the x-direction, but the errors are slightly different which is likely due to the different coefficients in the functions. Both methods are able to exactly reconstruct the surface gradient of the linear function.

However, in contrast to the tests on the equidistant quadrilateral mesh, the gradient of the bi-linear and quadratic functions cannot be reconstructed in an exact manner anymore. For both methods, the error of the non-exact surface gradient approximations reduce with approximately $\mathcal{O}(h^{1.5})$, where the Green-Gauss method performs slightly better regarding the sine/cosine function.

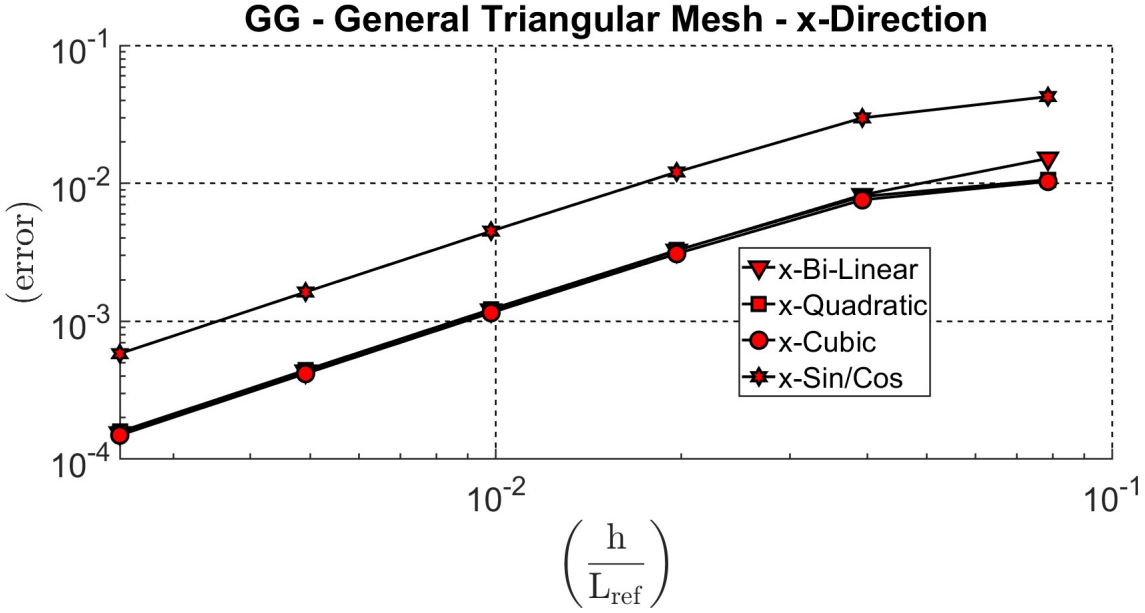


Figure 33: Surface gradient approximation error in the x-direction for the Green-Gauss method on a general triangular mesh as a function of the non-dimensional characteristic length.

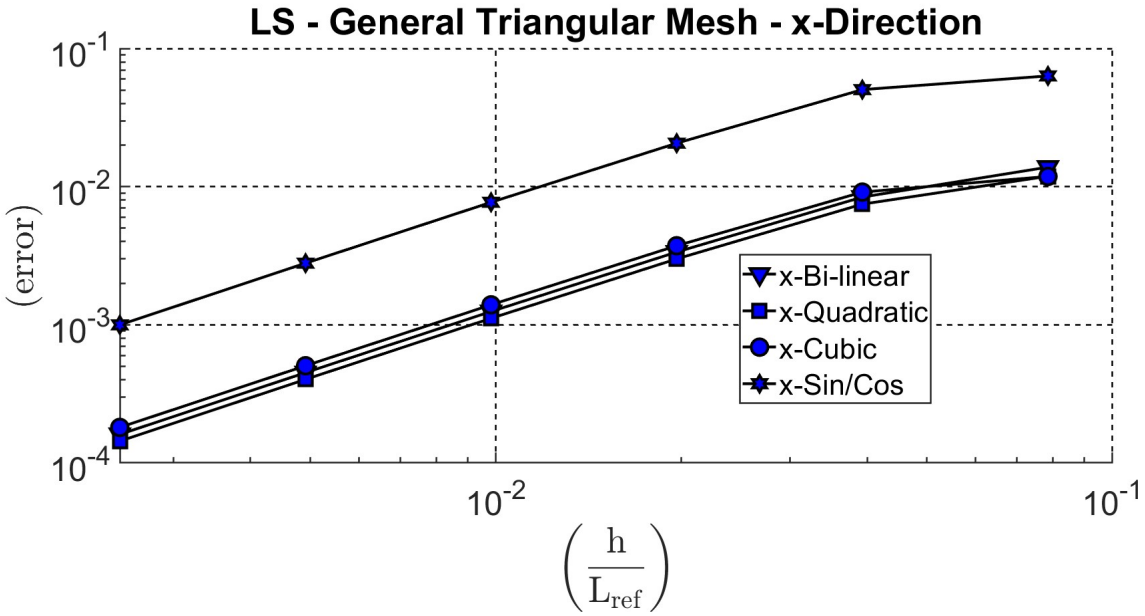


Figure 34: Surface gradient approximation error in the x-direction for the least squares method on a general triangular mesh as a function of the non-dimensional characteristic length.

In contrast to the equidistant quadrilateral mesh that has a similar secondary mesh and the general triangular mesh that produces a decent quality secondary mesh, the perturbed meshes as shown in Figures (24) and (25) produce secondary meshes of bad quality. Again, all function values are determined for these

meshes and the Green-Gauss method and the least squares method are used to reconstruct the surface gradients. For the perturbed quadrilateral mesh, the results related to the x-direction are depicted in Figures (35) and (36), while the results related to the y-direction are shown in Appendix (E). Similarly, the results related to the x-direction for the perturbed triangular grid are shown in Figures (37) and (38) and the corresponding results for the y-direction are placed in Appendix (E).

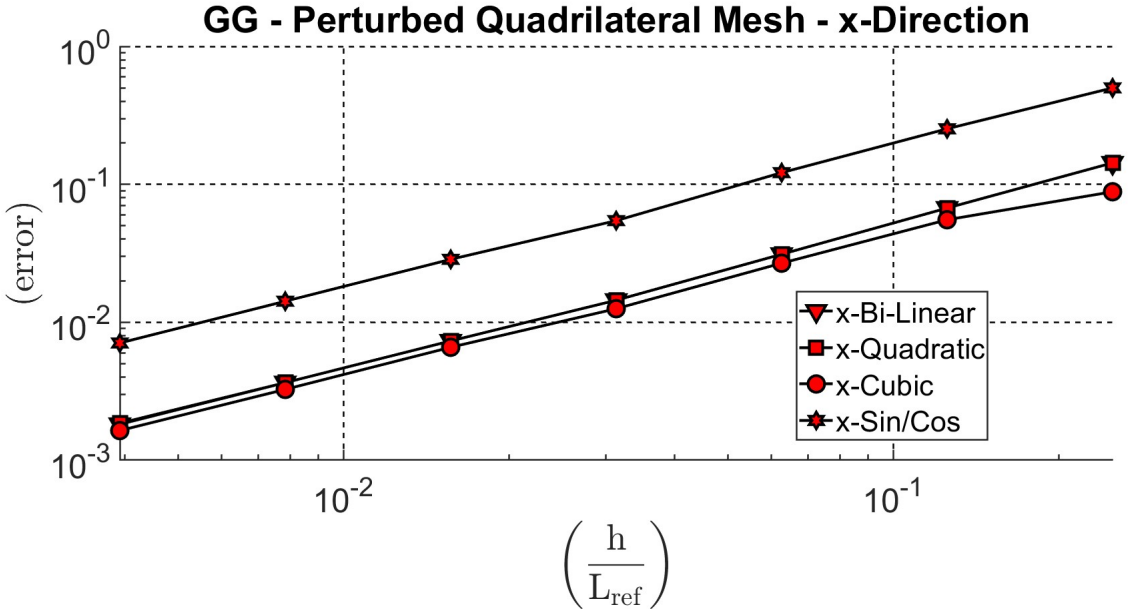


Figure 35: Surface gradient approximation error in the x-direction for the Green-Gauss method on a general perturbed quadrilateral mesh as a function of the non-dimensional characteristic length.

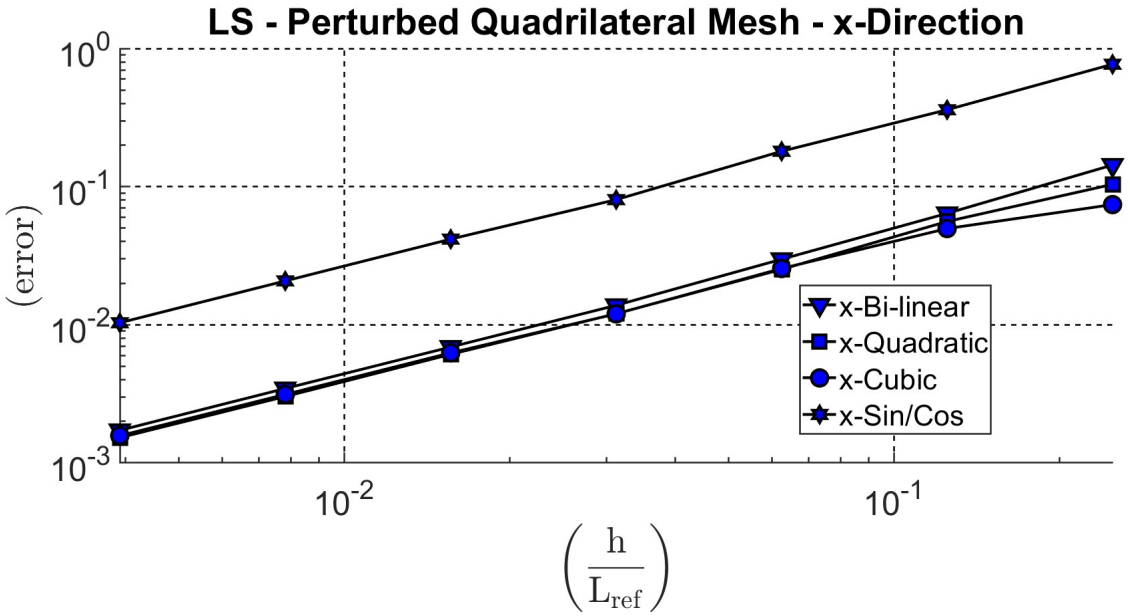


Figure 36: Surface gradient approximation error in the x-direction for the least squares method on a perturbed quadrilateral mesh as a function of the non-dimensional characteristic length.

Apart from minor shifts upwards or downwards, the graphs representing the results of the y-direction are similar to the results in x-direction. It should be noted that the error for the quadratic function depicted

in Figure (49) suddenly changes its path upon decreasing the non-dimensional characteristic length. The reason for this is unknown, but it is not further investigated as the higher order functions on the same mesh do not show this behaviour and the quadratic function itself does also not behave this way for the perturbed triangular mesh as depicted in Figure (51). For both methods, the results depicted in Figures (35), (36), (37) and (38) are quite similar as the error is of $\mathcal{O}(h)$. The Green-Gauss method produces slightly lower errors for the sine/cosine function and the least squares method works better for perturbed quadrilateral elements than perturbed triangular elements, but all these differences are minor.

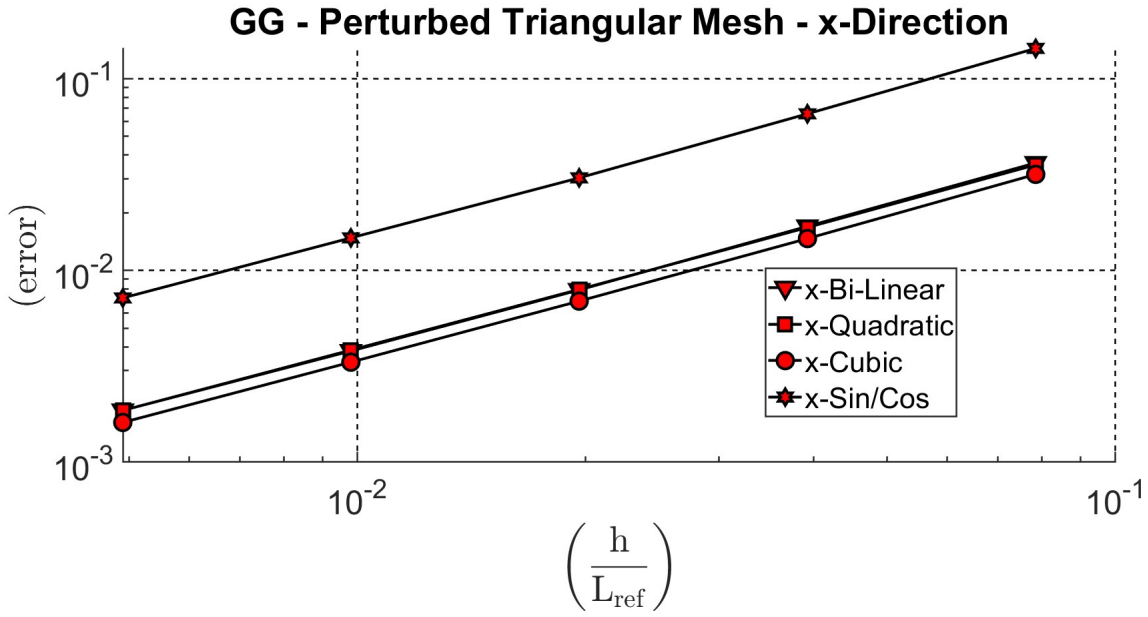


Figure 37: Surface gradient approximation error in the x-direction for the Green-Gauss method on a perturbed triangular mesh as a function of the non-dimensional characteristic length.

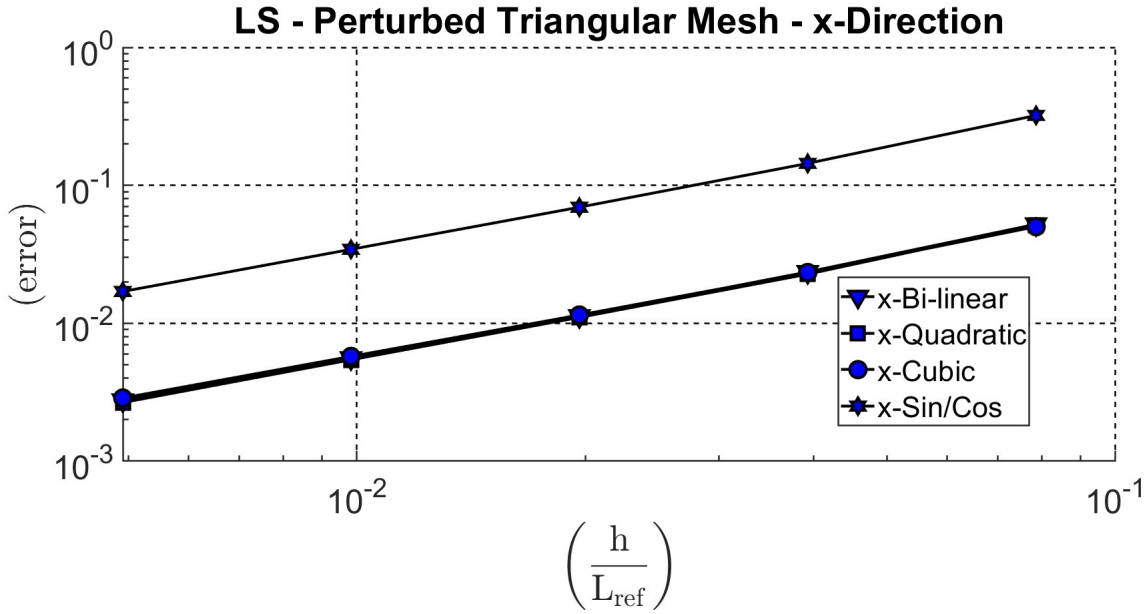


Figure 38: Surface gradient approximation error in the x-direction for the least squares method on a perturbed triangular mesh as a function of the non-dimensional characteristic length.

In general, when considering flat grids, both surface gradient reconstruction schemes are able to achieve

convergence, even for highly unstructured grids. Although the Green-Gauss method has slightly lower errors for the sine/cosine functions than the least squares method, there are hardly any significant differences. From these tests, in comparison with the benchmark test regarding the equidistant quadrilateral mesh, it can be seen that the use of unstructured meshes significantly increases the errors. For an arguably neatly formed unstructured grid (Figure (23)), the error is already more than a whole order of magnitude bigger than the error of the benchmark test regarding the cubic function, while the error is a whole order of magnitude bigger for the sine/cosine function. For the perturbed meshes, the error can be bigger by two orders of magnitude. This shows that the error is heavily dependent on the quality of the mesh.

4.3 Curved Mesh Test Cases

As observed previously, the Green-Gauss surface gradient reconstruction scheme is not exact anymore when the surface is approximated i.e. a curved surface. Therefore, as a start, a rectangular mesh is curved by letting the z-coordinates vary quadratically with the x-coordinates as depicted in Figures (26) and (27) for quadrilateral and triangular elements respectively. The corresponding errors are depicted in Figures (39) and (40), where $L_{ref} = 4$ is used. On both meshes, only the linear and the sine/cosine functions are tested. The linear function is used to confirm the inability of the Green-Gauss method to exactly represent the surface gradient of a linear function and the sine/cosine function as it requires an infinite amount of terms in the Taylor series to reconstruct it. On the quadrilateral and triangular mesh, the gradient of the linear function is indeed not exactly reconstructable. However, on both meshes, as well as for both functions, the error acts between $\mathcal{O}(h^{1.5})$ and $\mathcal{O}(h)$. However, when looking at the results for the quadrilateral mesh in Figure (39), for both functions, the y-direction starts to deviate from its normal order of descent. This is not expected, because the corresponding mesh shown in Figure (26) shows that the nodes and the cell centers in the y-direction are nicely aligned. A similar trend is observed for the linear function in the z-direction for the curved triangular mesh. This could be related to the effect of curvature, but the same is not observed for the sine/cosine function in z-direction. Additionally, for the curved triangular mesh, it can be seen that due to the bad initial grid quality asymptotic convergence takes longer. Finally, it can be observed that the y-direction for the sine/cosine function results in a slower convergence which again, is not in the direction of the curvature.

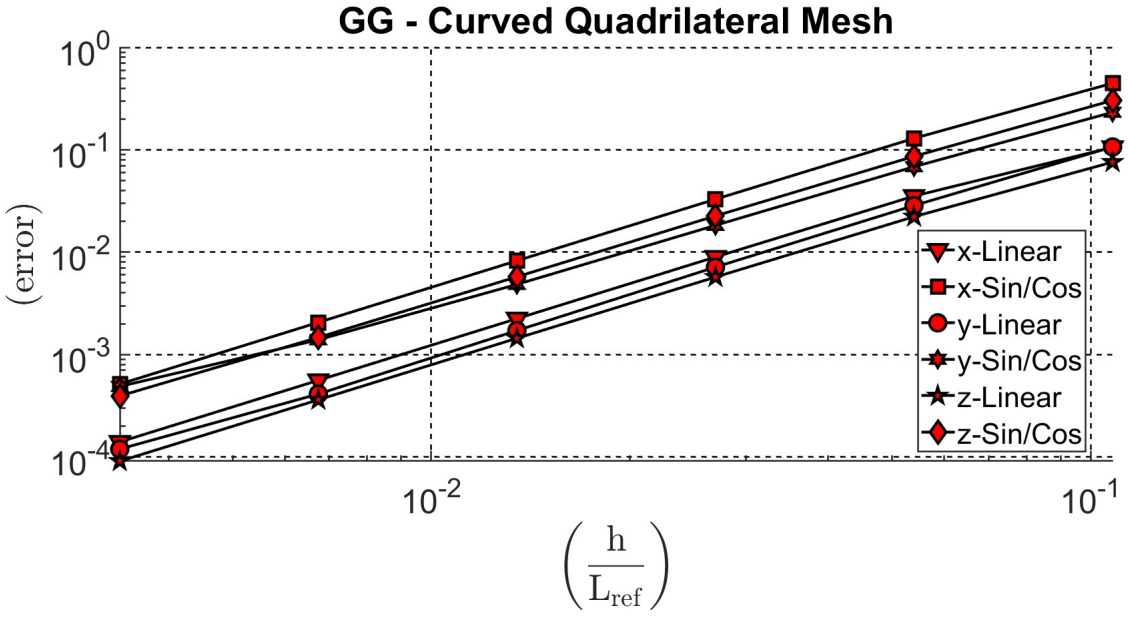


Figure 39: Surface gradient approximation error for the Green-Gauss method on a curved quadrilateral mesh as a function of the non-dimensional characteristic length.

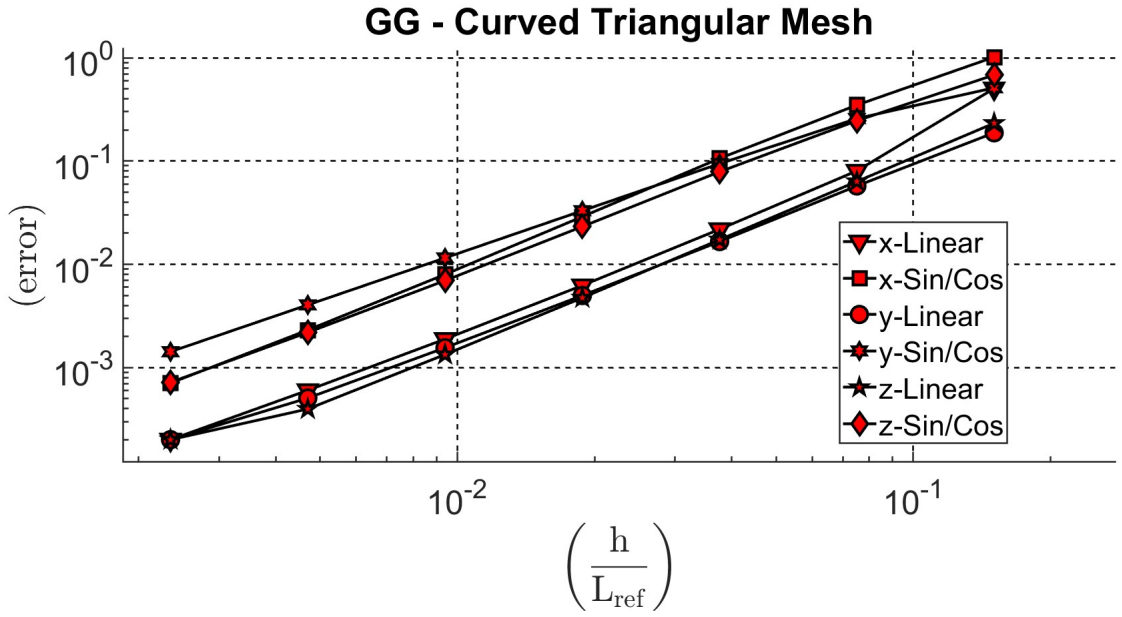


Figure 40: Surface gradient approximation error for the Green-Gauss method on a curved triangular mesh as a function of the non-dimensional characteristic length.

Though most of the errors show convergence with a decreasing characteristic length, the bigger error in the y-direction and the sudden decrease in convergence have no reasonable cause and explanation to the authors best knowledge. Finally, the function values are determined for the meshes shown in Figures (28) and (29). The reference length $L_{ref} = 10$ is used to determine the results of which the results in the x-direction are shown in Figures (41) and (42) for the quadrilateral mesh and the triangular mesh respectively. The results for the y- and z-direction for the triangular mesh are depicted in Figures (53) and (54) in Appendix (E) as these show similar results as the x-direction. First of all Figure (42), which shows that the Green-Gauss gradient reconstruction method results in an error of approximately $\mathcal{O}(h^{1.5})$ on a triangular mesh over a sphere for all functions tested. The results show that an increase in function order also results in a higher error, however, this does not hold for the cubic function as that results in a higher error than the sine/cosine function. In contrast to the results for the triangular mesh, the results for the quadrilateral mesh as shown in Figure (41) show no convergence, where for an even more refined grid also the sine/cosine function stops converging (this is not shown in the figure). This is definitely not expected as this mesh is of visibly better quality than the triangular grid. Additionally, despite the inexplicable behaviour of the results in y-direction for the curved quadrilateral mesh, the corresponding errors converged and did not show the same behaviour as for the mesh over the sphere. Moreover, the area convergence plot of Figure (30) shows that the mesh converges to the real surface. So, as an additional test, the unweighted least squares method was applied to both quadrilateral and triangular meshes to reconstruct the surface gradient that corresponds to the linear function. For both meshes, the least squares method is able to exactly reconstruct the surface gradient. Therefore, it is likely that a mistake is made with regards to the implementation of the Green-Gauss method or possibly the half-edge data structure as all other results suggest that the results for the quadrilateral mesh over a sphere should converge. Due to a lack of time, the author was not able to determine whether it is a mistake in the implementation in Fortran or if there is a different cause.

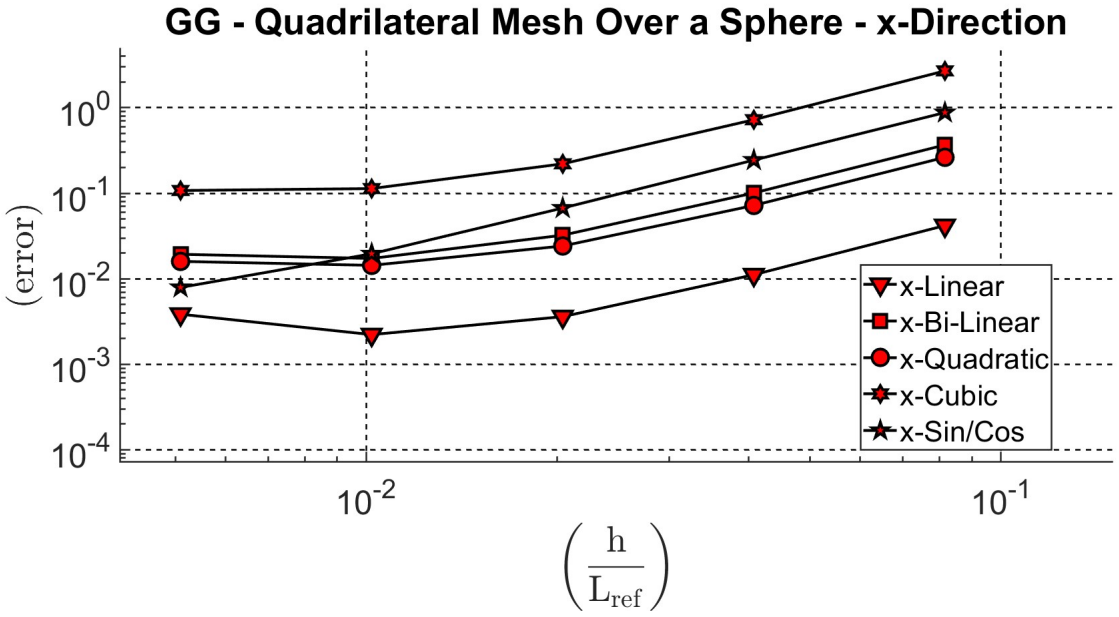


Figure 41: Surface gradient approximation error in the x-directions for the Green-Gauss method on a quadrilateral mesh over the surface of a sphere as a function of the non-dimensional characteristic length.

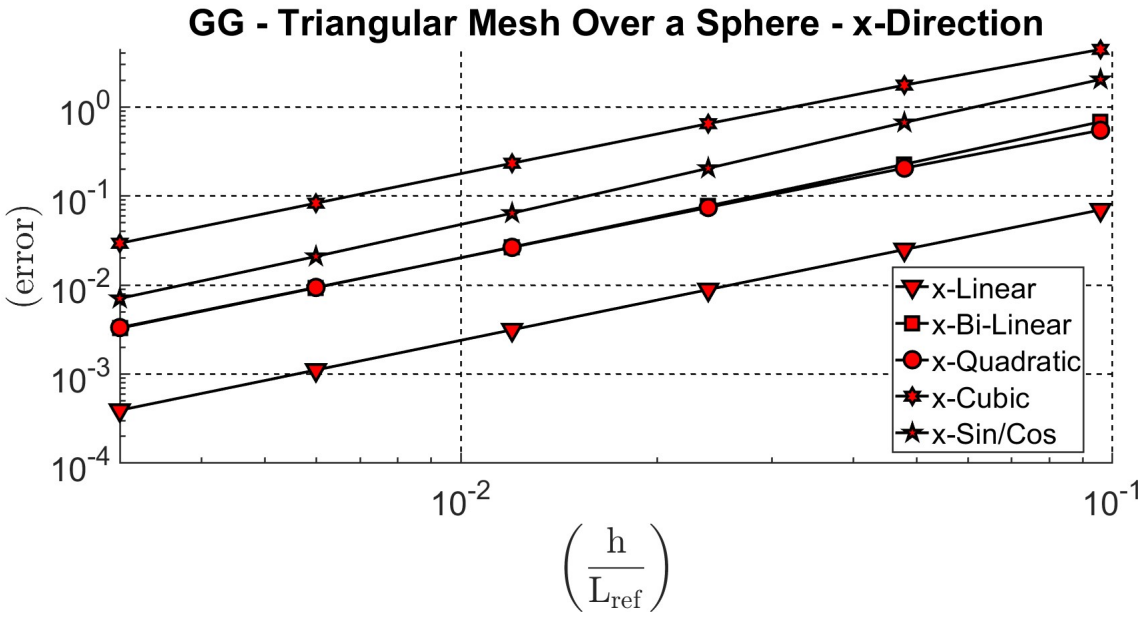


Figure 42: Surface gradient approximation error in the x-directions for the Green-Gauss method on a triangular mesh over the surface of a sphere as a function of the non-dimensional characteristic length.

5 Future Research and Conclusion

5.1 Future Research

Due to the inexplicable results and a plausible mistake with regards to the implementation of the Green-Gauss method, there is room for improvement. First of all, a complete re-implementation of the half-edge data structure and the Green-Gauss surface gradient reconstruction scheme could provide clarity as to whether there is a mistake in the implementation or if the cause should be sought elsewhere. To add

to that, the computational efficiency of the half-edge data structure could be compared with structured approaches or with other data structures, such as the quad-edge, split-edge or corner data structures. Furthermore, similar to much of the research into gradient reconstruction schemes in the finite volume method, more emphasis can be placed on the least squares method. Especially for unstructured meshes, the error can significantly decrease if appropriate weighting factors are used [20]. Apart from using weighting factors, the least squares method also has the advantage that extended stencils can be used to incorporate a bigger neighbourhood around each vertex. This can also be done in connection with further research into dealing with boundary nodes. Specifically for vertices on the trailing edge of a wing it is complicated to determine the surface gradient as there is only data from one direction.

5.2 Conclusion

The development of a hybrid grid advanced low-order panel method that is capable of handling structured and unstructured meshes was the driving factor behind this research. Although a significantly simplified version of a panel method was considered i.e. no wake surface and only a distribution of doublets, small steps have been made. First of all, the half-edge data structure is capable of navigating both structured and unstructured meshes and has the potential to be used for more applications, such as mesh subdivision, mesh simplification and mesh regularisation/smoothing [26]. Additionally, the Green-Gauss surface gradient reconstruction method and the least squares surface gradient reconstruction method provide hopeful results. For meshes without curvature, both methods are able to reconstruct the surface gradients up to $\mathcal{O}(h^2)$ with only highly unstructured perturbed meshes resulting in errors of $\mathcal{O}(h)$. For meshes over curved surfaces, similar trends are seen with converging results. However, no hard conclusions can be made with regards to those results as the quadrilateral mesh over a sphere results in a non-converging error.

References

[1] J. M. Ezquerro Navarro, "Panel Methods for the Aerodynamic Calculation of Mixed Configurations with Finite Thickness and Zero Thickness," *Universidad Politécnica de Madrid*, June 2017.

[2] J. Katz and A. Plotkin, *Low-Speed Aerodynamics*. New York: Cambridge University Press, 2001.

[3] M. Drela, "XFOIL: An Analysis and Design System for Low Reynolds Number Airfoils," *Lecture Notes in Engineering*, vol. 54, pp. 1–12, 1989.

[4] A. van Garrel, "Multilevel Panel Method for Wind Turbine Rotor Flow Simulations," *Thesis University of Twente*, 2016.

[5] M. Drela, "XFOIL Subsonic Airfoil Development System," https://web.mit.edu/drela/Public/web/xfoil/.

[6] N. Ramos García, J. N. Sørensen, and W. Z. Shen, "A Strong Viscous–Inviscid Interaction Model for Rotating Airfoils," *Wind Energy*, vol. 17, no. 12, pp. 1957–1984, 2014.

[7] A. van Garrel, "Development of a Wind Turbine Rotor Flow Panel Method," ECN-E–11-071, Energy research Centre of the Netherlands, 2011.

[8] E. Ortega, R. Flores, and E. Oñate, "A 3D Low-Order Panel Method for Unsteady Aerodynamic Problems," *International Center for Numerical Methods in Engineering*, May 2010.

[9] J. F. B. Pedro, "Boundary Elements Method for Three-Dimensional Potential Flow Based on Unstructured Meshes," 2008.

[10] C. R. Satterwhite, "Development of CPanel, an Unstructured Panel Code, Using a Modified TLS Velocity Formulation," *Faculty of California Polytechnic State University*, August 2015.

[11] A. Syrakos, S. Varchanis, Y. Dimakopoulos, A. Goulas, and J. Tsamopoulos, "A Critical Analysis of Some Popular Methods for the Discretisation of the Gradient Operator in Finite Volume Methods," *Physics of Fluids*, vol. 29, December 2017.

[12] E. Sozer, C. Brehm, and C. C. Kiris, "Gradient Calculation Methods on Arbitrary Polyhedral Unstructured Meshes for Cell-Centered CFD Solvers," *American Institute of Aeronautics and Astronautics*, January 2014.

[13] M. Deka, S. Brahmachary, R. Thirumalaisamy, A. Dalal, and G. Natarajan, "A New Green–Gauss Reconstruction on Unstructured Meshes. Part I: Gradient Reconstruction," *Journal of Computational Physics*, October 2018.

[14] S. S. C. Athkuri and V. Eswaran, "A New Auxiliary Volume-Based Gradient Algorithm for Triangular and Tetrahedral Meshes," *Journal of Computational Physics*, August 2020.

[15] H. Nishikawa, "From Hyperbolic Diffusion Scheme to Gradient Method: Implicit Green–Gauss Gradients for Unstructured Grids," *Journal of Computational Physics*, vol. 372, June 2018.

[16] A. Katz and V. Sankaran, "High Aspect Ratio Grid Effects on the Accuracy of Navier–Stokes Solutions on Unstructured Meshes," *Computers & Fluids*, vol. 65, March 2012.

[17] S. Seo, C. Lee, E. Kim, K. Yune, and C. Kim, "Least-Square Switching Process for Accurate and Efficient Gradient Estimation on Unstructured Grid," *Journal of the Korean Society for Industrial and Applied Mathematics*, vol. 24, March 2020.

[18] H. Jasak and H. G. Weller, "Application of the Finite Volume Method and Unstructured Meshes to Linear Elasticity," *International Journal for Numerical Methods in Engineering*, vol. 48, July 1999.

[19] D. J. Mavriplis, "Revisiting the Least-Squares Procedure for Gradient Reconstruction on Unstructured Meshes," *National Aeronautics and Space Administration*, December 2003.

[20] A. Syrakos, O. Oxtoby, E. de Villiers, S. Varchanis, Y. Dimakopoulos, and J. Tsamopoulos, "A Unification of Least-Squares and Green–Gauss Gradients Under a Common Projection-Based Gradient Reconstruction Framework," *Mathematics and Computers in Simulation*, vol. 205, September 2023.

[21] Y. Wei, F. Zhang, J. Liu, H. Su, and C. Xu, "A Constrained Boundary Gradient Reconstruction Method for Unstructured Finite Volume Discretization of the Euler Equations," *Computers and Fluids*, vol. 252, December 2023.

[22] Z. Chen, F. Zhang, J. Liu, and B. Chen, "An Iterative Near-Boundary Reconstruction Strategy for Unstructured Finite Volume Method," *Journal of Computational Physics*, vol. 418, June 2020.

[23] C. Mancinelli, M. Livesu, and E. Puppo, "A Comparison of Methods for Gradient Field Estimation on Simplicial Meshes," *Computers & Graphics*, vol. 80, March 2019.

[24] C. D. Correa, R. Hero, and K. L. Ma, "A Comparison of Gradient Estimation Methods for Volume Rendering on Unstructured Meshes," *Transactions on Visualization and Computers Graphics*, vol. 17, March 2011.

[25] J. W. Slater, "Examining spatial (grid) convergence," February 2021. [Online; accessed 20-June-2024].

[26] K. Fatahalian, "Mesh Representations and Geometry Processing," Stanford University, 2021.

[27] J. Legakis, "Adjacency Data Structures - Introduction," Massachusetts Institute of Technology, 1998.

[28] B. G. Baumgart, "Winged Edge Polyhedron Representation," *National Technical Information Service*, vol. 141, October 1972.

[29] M. McGuire, "The Half-Edge Data Structure," Flipcode.com, 2000.

[30] T. J. Alumbaugh and X. Jiao, "Compact Array-Based Mesh Data Structures," *Proceedings of the 14th International Meshing Roundtable*, pp. 485–503, 2005.

[31] RWTH-Aachen-University, "The Half-Edge Data Structure," OpenMesh.org.

[32] M. Buijs, "Weighted Vertex Normals," Bytehazard.com, 2007.

[33] J. D'Elia, M. Storti, and S. Idelshon, "Smoothed Surface Gradients for Panel Methods," *Advances in Engineering Software*, vol. 31, pp. 339–346, 2000.

[34] R. S. Subramanian, "Boundary Conditions in Fluid Mechanics," *Department of Chemical and Biomolecular Engineering - Clarkson University*.

[35] C. Gauzaine and J. F. Remacle, "Gmsh: A Three-Dimensional Finite Element Mesh Generator with Built-in Pre- and Post-Processing Facilities," *International Journal for Numerical Methods in Engineering*, vol. 79, no. 11, pp. 1309–1331, 2009.

[36] J. D. Anderson Jr., *Fundamentals of Aerodynamics*. New York: McGraw-Hill Education, 2017.

[37] M. Drela, *Flight Vehicle Aerodynamics*. London: The MIT Press, 2014.

[38] H. Lamb, *Hydrodynamics*. New York: Dover Publications, 1945.

[39] B. Maskew, "Prediction of Subsonic Aerodynamic Characteristics: A Case for Low-Order Panel Methods," *Journal of Aircraft*, February 1982.

[40] T. M. Lin, Y. C. Chuang, J. K. Lee, and C. A. Lin, "Parallelisation of Pressure Correction Method on Unstructured Grids," *Parallel Computational Fluid Dynamics*, pp. 451–458, 1999.

Appendices

A Conservation of Mass

Consider a control volume V that is stationary i.e. fixed in space, arbitrary in shape and bounded by a control surface S. The unit normal vector is defined to be positive when pointing out of the control volume. Over time the quantities within this control volume, such as the density ρ, will change due to the presence of a flow. A depiction of this control volume is shown in Figure (43). At a time t, the total amount of mass $M(t)$ within the control volume can be written as:

$$M(t) = \iiint_V \rho(\vec{x}, t)\, dV. \tag{48}$$

In a similar fashion, the total amount of mass at a time $t + \Delta t$ in the control volume can be written as:

$$M(t + \Delta t) = M(t) - \Delta t \iint_S \rho(\vec{x}, t)\vec{U} \cdot \bar{n}\, dS. \tag{49}$$

The total amount of mass present in the control volume at time t is only able to change due to mass flowing out of or flowing into the control volume through the control surface. As the normal vector is defined positive for pointing out of the control volume, a minus sign is required to determine the mass flowing into the control volume. However, the equation above can be rewritten as:

$$\frac{M(t + \Delta t) - M(t)}{\Delta t} + \iint_S \rho(\vec{x}, t)\vec{U} \cdot \bar{n}\, dS = 0. \tag{50}$$

To get a more accurate description of the change in the total amount of mass within control volume V, the following limit can be taken:

$$\lim_{\Delta t \to 0} \frac{M(t + \Delta t) - M(t)}{\Delta t} = \frac{\partial M}{\partial t}, \tag{51}$$

which results in the derivative of the total amount of mass with respect to t. Using the result of this limit in combination with the result of equation (48) and substituting this in equation (50) gives:

$$\frac{\partial}{\partial t} \iiint_V \rho(\vec{x}, t)\, dV + \iint_S \rho(\vec{x}, t)\vec{U} \cdot \bar{n}\, dS = 0. \tag{52}$$

This is a global statement about the control volume V and this equation is called the weak formulation as the density $\rho(\vec{x}, t)$ may be discontinuous (e.g. for shocks). By using Gauss' Divergence Theorem:

$$\iint_S \rho(\vec{x}, t)\vec{U} \cdot \bar{n}\, dS = \iiint_V \vec{\nabla} \cdot \left(\rho(\vec{x}, t)\vec{U}\right)\, dV, \tag{53}$$

and by noting that the control volume is fixed in space (stationary) i.e.

$$\frac{\partial}{\partial t} \iiint_V \rho(\vec{x}, t)\, dV = \iiint_V \frac{\partial \rho(\vec{x}, t)}{\partial t}\, dV, \tag{54}$$

equation (52) can be written as:

$$\iiint_V \left[\frac{\partial \rho(\vec{x}, t)}{\partial t} + \vec{\nabla} \cdot \left(\rho(\vec{x}, t)\vec{U}\right)\right]\, dV = 0. \tag{55}$$

In addition, as the control volume is of arbitrary shape, the terms within the integral must sum to zero which results in the strong formulation i.e.

$$\frac{\partial \rho(\vec{x},t)}{\partial t} + \vec{\nabla} \cdot \left(\rho(\vec{x},t)\vec{U} \right) = 0, \tag{56}$$

where it should be noted that $\rho(\vec{x},t)$ now must be continuous.

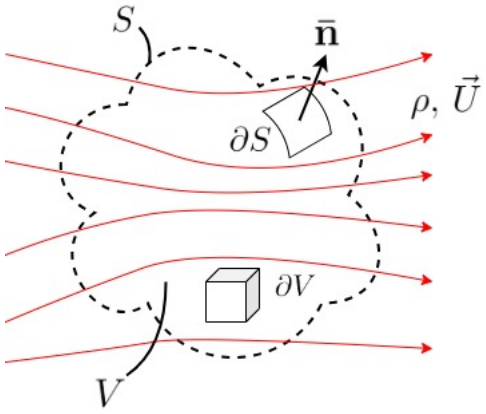


Figure 43: Arbitrary control volume V bounded by control surface S fixed in space (stationary).

By using the chain rule of differentiation and omitting the parentheses after ρ, this equation can be written as:

$$\frac{\partial \rho}{\partial t} + \rho \vec{\nabla} \cdot \vec{U} + \vec{U} \cdot \vec{\nabla} \rho = 0. \tag{57}$$

When making the assumption that the flow is incompressible (ρ = constant), the equation above simplifies significantly as all derivatives are zero and ρ itself can be divided out:

$$\vec{\nabla} \cdot \vec{U} = 0. \tag{58}$$

It should be noted that incompressible flows do not exist in nature. However, there are cases where incompressible flows are a good assumption such as liquids, which are hardly compressible. In addition, for low-speed flows of gas, in which the pressure differential is small, there is barely any change in density making the incompressible flow assumption acceptable. Finally, as a first rough estimate to many problems, the incompressible flow assumption can help to give insight towards a better understanding of the problem at hand [36], [37].

B Potential Flow

B.1 Potential Flow Problem Description

For flowfields characterised by high Reynolds numbers, the vorticity is confined to the boundary layer and wake regions in which the viscous effects cannot be neglected [2]. However, outside of the boundary layer and the wake region it is more appropriate to assume that the flow is inviscid and irrotational i.e. there is no vorticity:

$$\vec{\omega} = \vec{\nabla} \times \vec{U} = \vec{0}. \tag{59}$$

In addition, the curl of the gradient of a scalar function is always the zero vector field:

$$\vec{\nabla} \times (\vec{\nabla}\Phi) = \vec{0}. \tag{60}$$

So, under the assumption of an irrotational flow, it can be deduced from the equations above that there is always a potential function Φ of which the gradient is equal to the velocity vector:

$$\vec{U} = \vec{\nabla}\Phi. \tag{61}$$

From the assumption that the flow is also incompressible, the continuity equation can be reduced to

$$\vec{\nabla} \cdot \vec{U} = 0, \tag{62}$$

as discussed in Appendix (A). In turn, this can be written in terms of the velocity potential function Φ as:

$$\vec{\nabla} \cdot \vec{\nabla}\Phi = \nabla^2\Phi = 0. \tag{63}$$

This shows that the velocity potential function satisfies the Laplace equation. The Laplace equation is linear which means that superposition applies i.e. a combination of solutions is also a solution. Now, for a solid body with surface S_B, submerged in this incompressible, inviscid, irrotational flow, the continuity equation reduces to the equation above for which a solution is sought in the fluid domain. The fluid domain will be denoted by V and the domain itself can be enclosed by an outer flow boundary S_∞ as depicted in Figure (44). The flow itself should follow the contours of the body submerged in the flow to avoid penetration of the body by the fluid, but as viscosity is neglected the no-slip boundary condition is not available. However, by demanding the velocity component normal to the body's surface and other solid surfaces to be zero, the flow is forced to follow the contours of the boundaries. By fixing the coordinate system to the body this boundary condition can be denoted as:

$$\vec{\nabla}\Phi \cdot \bar{n} = 0. \tag{64}$$

Additionally, the presence of the body in the flow will cause for a disturbance which eventually decays. So, sufficiently far away from the body, the disturbance by the body may not be felt which can be described as:

$$\lim_{r\to\infty} \vec{\nabla}\Phi = \vec{u}_\infty. \tag{65}$$

Here, $\vec{u}_\infty$ describes the velocity of the undisturbed fluid in the inertial frame of reference and r denotes the distance from the body [2].

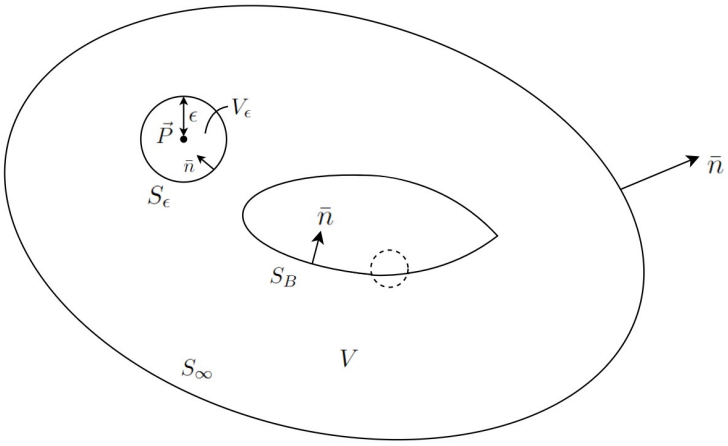


Figure 44: Schematic representation of the flow domain V in which a body with surface S_B is submerged. Additionally, point $\vec{P}$ is excluded by a sphere of surface S_ϵ.

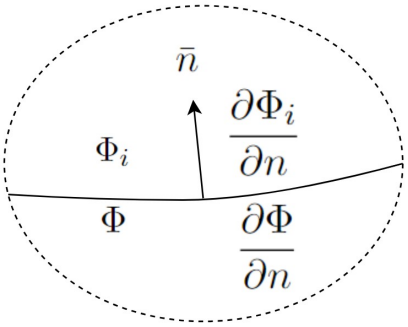


Figure 45: Enlarged depiction of the boundary S_B where the difference of the normal derivatives of the velocity potential and the velocity potentials are shown on the internal and external side of S_B.

B.2 General Solution of the Potential Flow Problem

A derivation of a general solution to the Laplace equation with the use of Green's second identity is discussed in Appendix (D.1). By not considering a wake, and therefore a wake surface, the total potential for a point $\vec{P}$ in a fluid domain V can be denoted as:

$$\Phi(\vec{P}) = \underbrace{-\frac{1}{4\pi}\iint\limits_{S_B}\left[\sigma\left(\frac{1}{r}\right)\right]dS}_{\phi_\sigma(\vec{P})} + \underbrace{\frac{1}{4\pi}\iint\limits_{S_B}\left[\mu\bar{n}\cdot\vec{\nabla}\left(\frac{1}{r}\right)\right]dS}_{\phi_\mu(\vec{P})} + \Phi_\infty(\vec{P}) \quad \text{when} \quad \vec{P}\in V. \tag{66}$$

Here, the total potential is a summation of the potential of the undisturbed fluid ($\Phi_\infty(\vec{P})$) plus two perturbation potentials due to the distribution of sources ($\phi_\sigma(\vec{P})$) and doublets ($\phi_\mu(\vec{P})$). These perturbation potentials are depending on the values of $\sigma(\vec{Q})$ and $\mu(\vec{Q})$ on the surface of the body S_B which are known as the source and doublet strengths respectively. The definition of these strengths are:

$$-\mu(\vec{Q}) = \Phi - \Phi_i, \tag{67}$$

$$-\sigma(\vec{Q}) = \frac{\partial\Phi}{\partial n} - \frac{\partial\Phi_i}{\partial n}. \tag{68}$$

In these formulations, $\vec{Q}$ denotes a point on the boundary and r is the distance between points $\vec{P}$ and $\vec{Q}$. Moreover, Φ and Φ_i, both functions of $\vec{Q}$, denote the value of the potential functions on the outside and inside of the surface respectively which can be seen in Figure (45). Additionally, the unit normal vector $\bar{n}$ is defined as shown in Figure (44).

B.3 Application of the Boundary Conditions

As discussed in Appendix (D.1), Equation (66) does not represent a unique solution as the distribution of the singularity elements i.e. sources and doublets, must be specified on the surface which can be done with the help of a boundary condition. First of all, by considering the boundary condition given in Equation (65) and taking the gradient of Equation (66):

$$\vec{\nabla}\Phi = -\frac{1}{4\pi}\iint\limits_{S_B}\sigma\vec{\nabla}\left(\frac{1}{r}\right)dS + \frac{1}{4\pi}\iint\limits_{S_B}\mu\vec{\nabla}\left[\frac{\partial}{\partial n}\left(\frac{1}{r}\right)\right]dS + \vec{\nabla}\Phi_\infty, \tag{69}$$

it can be seen that when $r \to \infty$, the integrals vanish and the potential Φ_∞ results in the velocity $\vec{u}_\infty$. As a result, this boundary condition is automatically satisfied as both the source and doublet solutions decay for $r \to \infty$ [2]. The boundary condition given in Equation (64) is a Neumann boundary condition that states that the normal derivative of the potential function is zero on the boundaries. Lamb [38] observed that, if this holds for an enclosed boundary such as S_B, the total internal potential Φ_i is not changed and therefore, this Neumann boundary condition can be written as a Dirichlet boundary condition:

$$\Phi_i = \text{constant}, \tag{70}$$

where the constant is free to choose. So, for a point $\vec{P}$ inside body S_B, the potential function can be written as:

$$\Phi_i(\vec{P}) = \phi_\sigma(\vec{P}) + \phi_\mu(\vec{P}) + \Phi_\infty(\vec{P}) = \text{constant}. \tag{71}$$

By setting $\Phi_i = 0$, the equation above can be rewritten as:

$$\phi_\sigma(\vec{P}) + \phi_\mu(\vec{P}) = -\Phi_\infty(\vec{P}). \tag{72}$$

Since the internal potential is constant, also $\Phi_i = 0$ which simplifies the equations for the doublet and source strengths. Using Equation (67), the potential on the outside of S_B can now be written as:

$$\Phi = -\mu. \tag{73}$$

Similarly, Equation (68) is simplified to:

$$\sigma = -\frac{\partial \Phi}{\partial n}, \tag{74}$$

where the normal derivative on the outside of the surface must be equal to the velocity of the surface in the normal direction plus a specified outflow velocity (that can be used to simulate in- or outflows, boundary layer displacement effects etc. [39], [4]). Therefore, this equation can be rewritten as:

$$\sigma = -u_n + \bar{n} \cdot \vec{U}_S. \tag{75}$$

Arguably, the simplest problem can be obtained by assuming that both the specified outflow velocity and the surface velocity are zero. Consequently, $\sigma = 0$, meaning that the problem contains no source distributions, but only doublet distributions. As a result, Equation (72) now becomes:

$$\phi_\mu(\vec{P}) = -\Phi_\infty(\vec{P}). \tag{76}$$

With the help of a panel method, this equation can solve for the doublet strengths μ. This is necessary, as the potential on the outside of the surface given in Equation (73), is purely dependent on the doublet strength. However, it is not the potential that is of interest, but the gradient of the potential as this gives the fluid flow velocity on the outside of S_B. Since the normal component of this gradient is zero, it is possible to take the surface gradient of both sides of Equation (73) which gives an equation for the velocity components tangential to the surface:

$$\vec{\nabla}_S \Phi = -\vec{\nabla}_S \mu. \tag{77}$$

B.4 Determining the Pressure Coefficients

If the values for the doublet strengths are known, the tangential velocity components of the flow can be determined. Since the flow is assumed to be inviscid, irrotational, incompressible and steady, Bernoulli's equation holds:

$$\frac{1}{2}\rho_\infty U_\infty^2 + p_\infty = \frac{1}{2}\rho U^2 + p, \tag{78}$$

in which ρ_∞, p_∞ and U_∞ are the density, pressure and velocity of the undisturbed free stream flow. Similarly, ρ, p and U denote the density, pressure and velocity at the point of interest. It should be noted that gravity is neglected here. This equation can be rewritten as:

$$p - p_\infty = \frac{1}{2}\rho_\infty U_\infty^2 - \frac{1}{2}\rho_\infty U^2, \tag{79}$$

where ρ is now replaced with ρ_∞ as the density is constant in incompressible flows. Additionally, the dimensionless pressure coefficient is defined as:

$$C_p = \frac{p - p_\infty}{\frac{1}{2}\rho_\infty U_\infty^2}. \tag{80}$$

Upon combining both equations, an expression for the pressure coefficient at each panel is obtained:

$$C_{p_k} = 1 - \frac{U_k^2}{U_\infty^2}, \tag{81}$$

in which U_k denotes the total velocity in the local coordinates of panel k i.e. the velocities obtained from Equation (77) plus the influence of the free stream velocity in the local panel coordinates.

C Low-Order Panel Method

As discussed in Appendix (B), the potential flow model reduced the problem in a fluid domain V to a problem at the boundaries of the domain. By placing a unique selection of singularity elements over the surface, which have to comply with the boundary conditions, it is possible to determine the tangential flow velocity components. In the case described before, these components are dependent on the surface gradient of the doublet strengths. However, the doublet strengths are still unknown and the distribution of these doublets is also not known yet.

C.1 Discretization of the Integral Equation

The distribution of the doublets over the surface is done by discretising the surface into $k = 1, 2, ..., N$ planar panels as shown in Figure (46). Similar to D'Elia et al. [33] and Pedro [9], these panels can be any convex polygonal shape, together forming an unstructured mesh over the surface. The benefit in the use of unstructured meshes lies in the ability to use automatic grid generators as grid generation is a very time-consuming part in numerical methods. Additionally, unstructured grids are able to cope with complex arbitrary geometries and are easier with regards to local mesh refinements [40]. However, this comes at a cost of increased complexity regarding the navigation of the mesh (data structure) and accuracy. Moving on, on every panel, a singularity element of constant strength is placed, so the whole panel represents a single singularity strength value. Due to the fact that the panels are planar and not curved and that the singularity strengths are of constant value, this panel method is low order. It is possible to use higher order (curved) panels, but also to increase the order of the singularity strength distribution (linear or quadratic etc.) over a panel. As the internal potential is specified everywhere inside the body, so-called collocation points are used to specify the boundary conditions. These points are usually placed at a point beneath the panel. By discretising Equation (76), the system to solve now becomes:

$$\sum_{k=1}^{N} \frac{1}{4\pi} \int_{\text{panel}} \mu \bar{n} \cdot \vec{\nabla} \left(\frac{1}{r} \right) \, dS = -\Phi_{\infty,k}. \tag{82}$$

So, for each collocation point $\vec{P}$, the sum of the influences of all N panels is required. For a constant strength doublet element, the only unknowns are the doublet strengths themselves as the influence coefficients are only dependent on the panel geometries as can be seen in Figure (46). Therefore, the following system of equations can be written:

$$\begin{bmatrix} c_{11} & c_{12} & \dots & c_{1N} \\ c_{21} & c_{22} & \dots & c_{2N} \\ \vdots & \vdots & & \vdots \\ c_{N1} & c_{N2} & \dots & c_{NN} \end{bmatrix} \begin{Bmatrix} \mu_1 \\ \mu_2 \\ \vdots \\ \mu_N \end{Bmatrix} = \begin{Bmatrix} -\Phi_{\infty,1} \\ -\Phi_{\infty,2} \\ \vdots \\ -\Phi_{\infty,N} \end{Bmatrix}. \tag{83}$$

Here, coefficient c_{12} denotes the influence panel 2 has on the collocation point placed beneath panel 1. Solving this full matrix equation will result in the values for the doublet strengths.

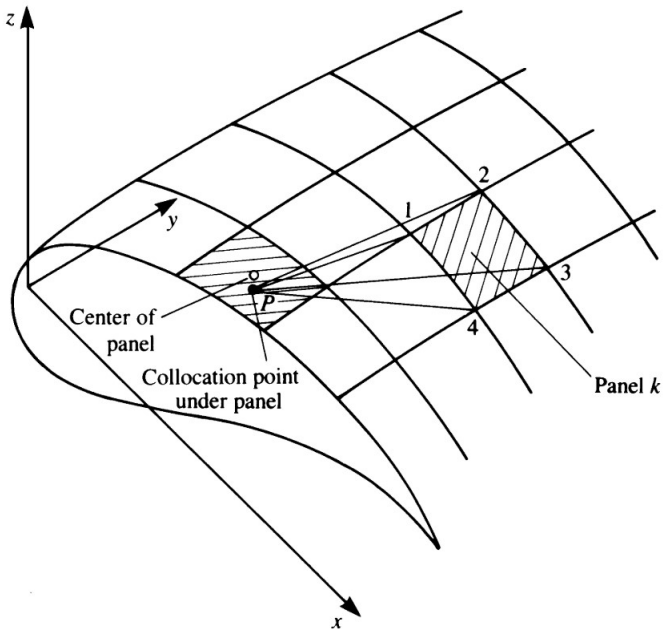


Figure 46: Depiction of the surface panelling and the influence of a panel k on the collocation point $\vec{P}$ [2].

D Boundary Integral Equation

D.1 Derivation of the Boundary Integral Equation

In a volume V, enclosed by surface S_∞, a body with surface S_B is submerged in an incompressible, inviscid, irrotational flow. The normal vector $\bar{n}$ is defined to always point out of the volume V. These definitions are depicted in Figures (44) and (45). It is conventional to also add a wake surface, but this is disregarded as it simplifies the problem and it is not the main focus of this research. As discussed in Appendix (B.1), the continuity equation reduces to the Laplace equation for a scalar potential function Φ. A general solution that is based on Green's second identity makes the definition of the velocity vector, as defined in Equation (61), slightly more complex as it defines the velocity vector as:

$$\vec{U} = \Phi_1 \vec{\nabla} \Phi_2 - \Phi_2 \vec{\nabla} \Phi_1 \tag{84}$$

In this definition, Φ_1 and Φ_2 are two position dependent scalar functions that are both continuously differentiable twice. Upon substitution of this definition into Gauss' Divergence Theorem, the following is obtained:

$$\iint_S (\Phi_1 \vec{\nabla} \Phi_2 - \Phi_2 \vec{\nabla} \Phi_1) \cdot \bar{n} \, dS = \iiint_V \left(\Phi_1 \nabla^2 \Phi_2 - \Phi_2 \nabla^2 \Phi_1 \right) \, dV. \tag{85}$$

As discussed before, $\bar{n}$ points out of the volume V and $S = S_B + S_\infty$. Note that both terms in the volume integral contain the Laplace equation which is known to be zero for the flow considered here. In addition, as long as the two scalar functions are continuously differentiable twice, they can be specified. So, the functions will be set as:

$$\Phi_1 = \frac{1}{r} \quad \text{and} \quad \Phi_2 = \Phi. \tag{86}$$

Φ here denotes the potential function that describes the flow of interest in V at a fixed point $P(x, y, z)$. From that same point, r denotes the distance to a point $Q(x, y, z)$ on the surface [2], [4]. For point

$\vec{P}$ outside the flow domain V, both functions satisfy the Laplace equation (see Appendix (D.2)). As a consequence, Equation (85) now becomes

$$\iint_S \left[\frac{1}{r}\vec{\nabla}\Phi - \Phi\vec{\nabla}\left(\frac{1}{r}\right)\right]\cdot\bar{n}\,dS = 0, \quad \text{when} \quad \vec{P}\notin V. \tag{87}$$

A more interesting place for point $\vec{P}$ would be inside the flow domain V, but therefore, it must be excluded from the region of integration. This is done by surrounding the point by a very small sphere with radius ϵ and surface S_ϵ. Consequently, the integral equation now becomes

$$\iint_S \left[\frac{1}{r}\vec{\nabla}\Phi - \Phi\vec{\nabla}\left(\frac{1}{r}\right)\right]\cdot\bar{n}\,dS + \iint_{S_\epsilon} \left[\frac{1}{r}\vec{\nabla}\Phi - \Phi\vec{\nabla}\left(\frac{1}{r}\right)\right]\cdot\bar{n}\,dS = 0 \quad \text{when} \quad \vec{P}\in V. \tag{88}$$

For the integral over the surface of the sphere, it is convenient to introduce a spherical coordinate system with point $\vec{P}$ as its origin. As a result, the dot product between the gradients and the normal vector can be written differently. The normal points into the sphere and is purely radial i.e. $\bar{n} = -\bar{e}_r$. As a result, the dot product only provides values in the radial direction:

$$\iint_{S_\epsilon} \left[\frac{1}{r}\vec{\nabla}\Phi - \Phi\vec{\nabla}\left(\frac{1}{r}\right)\right]\cdot\bar{n}\,dS = -\iint_{S_\epsilon} \left[\frac{1}{r}\frac{\partial\Phi}{\partial r} + \frac{\Phi}{r^2}\right]\,dS. \tag{89}$$

When $\epsilon \to 0$ and assuming the potential function and its derivatives do not vary much within the sphere, the first term in this integral vanishes. Moreover, the surface area of a sphere with $r = \epsilon$ is known, so the integral above results in:

$$-\iint_{S_\epsilon} \left[\frac{1}{r}\frac{\partial\Phi}{\partial r} + \frac{\Phi}{r^2}\right]\,dS \approx -\iint_{S_\epsilon} \left[\frac{\Phi}{r^2}\right]\,dS = -4\pi\Phi(\vec{P}). \tag{90}$$

Substitution of this result in Equation (88) and rewriting it as a function of $\Phi(\vec{P})$ gives

$$\Phi(\vec{P}) = \frac{1}{4\pi}\iint_S \left[\frac{1}{r}\vec{\nabla}\Phi - \Phi\vec{\nabla}\left(\frac{1}{r}\right)\right]\cdot\bar{n}\,dS \quad \text{when} \quad \vec{P}\in V. \tag{91}$$

Similarly as for point $\vec{P}$ being inside the flow domain V, when it lies on the boundary, only half a sphere is required to exclude the point from the flow domain. The same approach will then result in

$$\Phi(\vec{P}) = \frac{1}{2\pi}\iint_S \left[\frac{1}{r}\vec{\nabla}\Phi - \Phi\vec{\nabla}\left(\frac{1}{r}\right)\right]\cdot\bar{n}\,dS \quad \text{when} \quad \vec{P}\in S. \tag{92}$$

Apart from an interest in a flow in V, it is also possible that this flow occurs inside S_B of which the resulting potential is Φ_i. If point $\vec{P}$ is in V, then it is not affected by the internal potential flow i.e.

$$\frac{1}{4\pi}\iint_{S_B} \left[\Phi_i\vec{\nabla}\left(\frac{1}{r}\right) - \frac{1}{r}\vec{\nabla}\Phi_i\right]\cdot\bar{n}\,dS = 0 \quad \text{when} \quad \vec{P}\in V. \tag{93}$$

Note that the direction of the normal does influence the signs in this equation and that this integral only includes the surface of the body S_B and not S_∞. By adding this equation to Equation (91), the following is obtained:

$$\Phi(\vec{P}) = \frac{1}{4\pi}\iint_{S_B} \left[\frac{1}{r}\vec{\nabla}(\Phi-\Phi_i) - (\Phi-\Phi_i)\vec{\nabla}\left(\frac{1}{r}\right)\right]\cdot\bar{n}\,dS + \frac{1}{4\pi}\iint_{S_\infty} \left[\frac{1}{r}\vec{\nabla}\Phi - \Phi\vec{\nabla}\left(\frac{1}{r}\right)\right]\cdot\bar{n}\,dS, \tag{94}$$

where $\vec{P} \in V$. The contribution from the integration over the boundary S_∞ frequently depends on the coordinate system making it a constant in, for example, a system where the body is moving through an undisturbed fluid [2]. However, it is also possible to state that by taking the limit of the distance between S_∞ and S_B to go to infinity, the contribution to $\Phi(\vec{P})$ from the integral over S_∞ is that of the unperturbed velocity potential $\Phi_\infty(\vec{P})$ [4]. As a result, the final boundary integral equation can be written as:

$$\Phi(\vec{P}) = \frac{1}{4\pi} \iint_{S_B} \left[\frac{1}{r} \vec{\nabla}(\Phi - \Phi_i) - (\Phi - \Phi_i) \vec{\nabla} \left(\frac{1}{r} \right) \right] \cdot \bar{n} \, dS + \Phi_\infty(\vec{P}) \quad \text{when} \quad \vec{P} \in V. \tag{95}$$

This equation shows that the velocity potential at any point of interest i.e. $\vec{P}$ in the volume can be determined by a function of the perturbation potentials and the normal derivative of the perturbation potentials at the boundaries [10]. The main problem now is to determine the values of these properties on the boundaries (see Figure (45)). When considering the boundary S_B, a new definition can be introduced for the difference in potentials internally and externally i.e.

$$-\mu = \Phi - \Phi_i. \tag{96}$$

Furthermore, another new definition can be introduced for the difference between the normal derivatives of the potentials internally and externally i.e.

$$-\sigma = \frac{\partial \Phi}{\partial n} - \frac{\partial \Phi_i}{\partial n}. \tag{97}$$

Again, the definition of the normal causes a minus sign in these definitions. These elements are well-known in potential theory as μ is known as a doublet element and σ is known as a source element. Substitution of these definitions results in the boundary integral equation:

$$\Phi(\vec{P}) = \underbrace{-\frac{1}{4\pi} \iint_{S_B} \left[\sigma \left(\frac{1}{r} \right) \right] dS}_{\phi_\sigma(\vec{P})} + \underbrace{\frac{1}{4\pi} \iint_{S_B} \left[\mu \bar{n} \cdot \vec{\nabla} \left(\frac{1}{r} \right) \right] dS}_{\phi_\mu(\vec{P})} + \Phi_\infty(\vec{P}) \quad \text{when} \quad \vec{P} \in V. \tag{98}$$

Finally, this equation can be further abbreviated to:

$$\Phi(\vec{P}) = \phi_\sigma(\vec{P}) + \phi_\mu(\vec{P}) + \Phi_\infty(\vec{P}) \quad \text{when} \quad \vec{P} \in V. \tag{99}$$

This equation shows that the total potential in a fluid domain of interest V at a point $\vec{P}$, is the free stream potential Φ_∞ plus two perturbation terms due to a distribution of sources σ and doublets μ on the surface. There are many combinations with regards to the distribution of these singularity elements and therefore, a choice must be made as the current formulation does not uniquely describe the solution. This specification of the distribution of the singularity elements can be deduced from the boundary condition that demands that the velocity components normal to the body's surface are zero .

D.2 The Laplace Equation in Spherical Coordinates

For irrotational, incompressible flow, the flow field can be described as the Laplacian of the velocity potential Φ. In a three dimensional Cartesian coordinate system, this equation can be written as:

$$\nabla^2 \Phi = \frac{\partial^2 \Phi}{\partial x^2} + \frac{\partial^2 \Phi}{\partial y^2} + \frac{\partial^2 \Phi}{\partial z^2} = 0. \tag{100}$$

In the derivation of the boundary integral equation, it is convenient to introduce a spherical coordinate system. Therefore, one can introduce the following relations between the Cartesian coordinate system and the spherical coordinate system:

$$x = r\sin(\theta)\cos(\phi), \quad y = r\sin(\theta)\sin(\phi) \quad \text{and} \quad z = r\cos(\theta), \tag{101}$$

in which r denotes the radius, θ denotes the inclination or the polar angle and ϕ denotes the azimuth or azimuthal angle. Conversely, one can write:

$$r = \sqrt{x^2 + y^2 + z^2}, \quad \theta = \arccos\left(\frac{z}{r}\right) \quad \text{and} \quad \phi = \arctan\left(\frac{y}{x}\right). \tag{102}$$

Using these coordinate transformations it is possible to write the Laplace equation in spherical coordinates. This can be done using the chain rule and a rather lengthy derivation. The gradient of the potential function in spherical coordinates can be written as:

$$\vec{\nabla}\Phi = \frac{\partial\Phi}{\partial r}\bar{r} + \frac{1}{r}\frac{\partial\Phi}{\partial\theta}\bar{\theta} + \frac{1}{r\sin(\theta)}\frac{\partial\Phi}{\partial\phi}\bar{\phi}, \tag{103}$$

while the Laplace equation in spherical coordinates can be written as:

$$\nabla^2\Phi = \frac{1}{r^2}\frac{\partial}{\partial r}\left(r^2\frac{\partial\Phi}{\partial r}\right) + \frac{1}{r^2\sin(\theta)}\frac{\partial}{\partial\theta}\left(\sin(\theta)\frac{\partial\Phi}{\partial\theta}\right) + \frac{1}{r^2\sin^2(\theta)}\frac{\partial\Phi}{\partial\phi} = 0. \tag{104}$$

In the derivation of the boundary integral equation, one of the potential functions is defined as:

$$\Phi = \frac{1}{r} \quad \text{with} \quad r \neq 0. \tag{105}$$

The first derivative i.e. the gradient of this function results in:

$$\vec{\nabla}\left(\frac{1}{r}\right) = -\frac{1}{r^2}\bar{r} \quad \text{with} \quad r \neq 0, \tag{106}$$

and similarly, the Laplacian of this definition of the potential function will result in:

$$\nabla^2\left(\frac{1}{r}\right) = 0 \quad \text{with} \quad r \neq 0. \tag{107}$$

This shows that this function, similar to the velocity potential in irrotational, incompressible flows, always satisfies the Laplace equation.

E Additional Figures

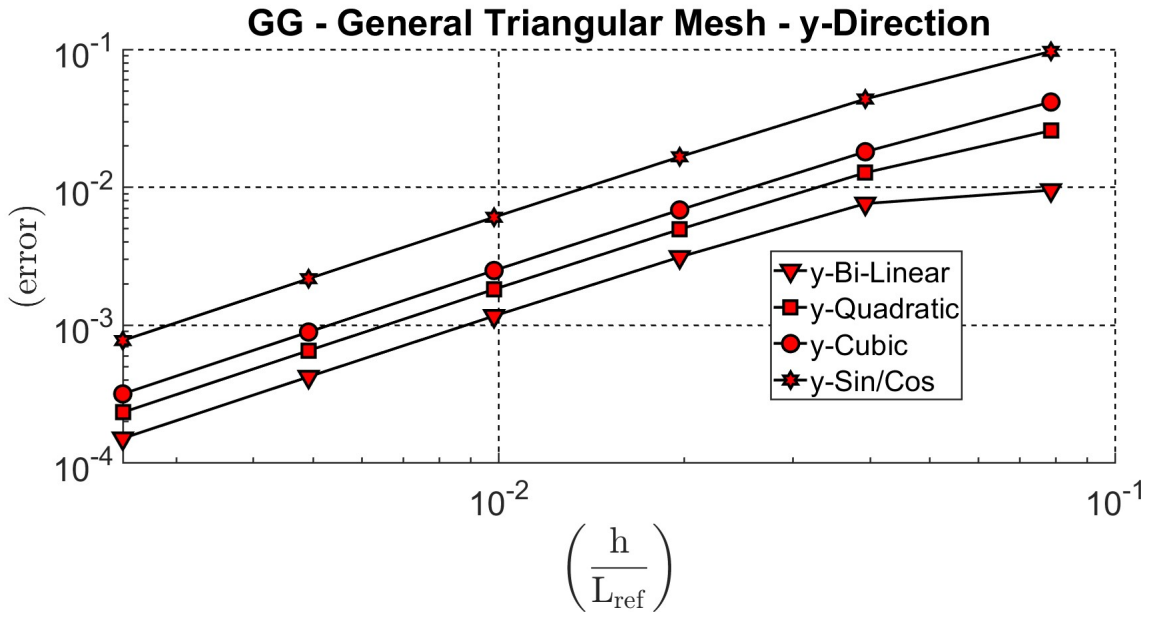


Figure 47: Surface gradient approximation error in the y-direction for the Green-Gauss method on a general triangular mesh as a function of the non-dimensional characteristic length.

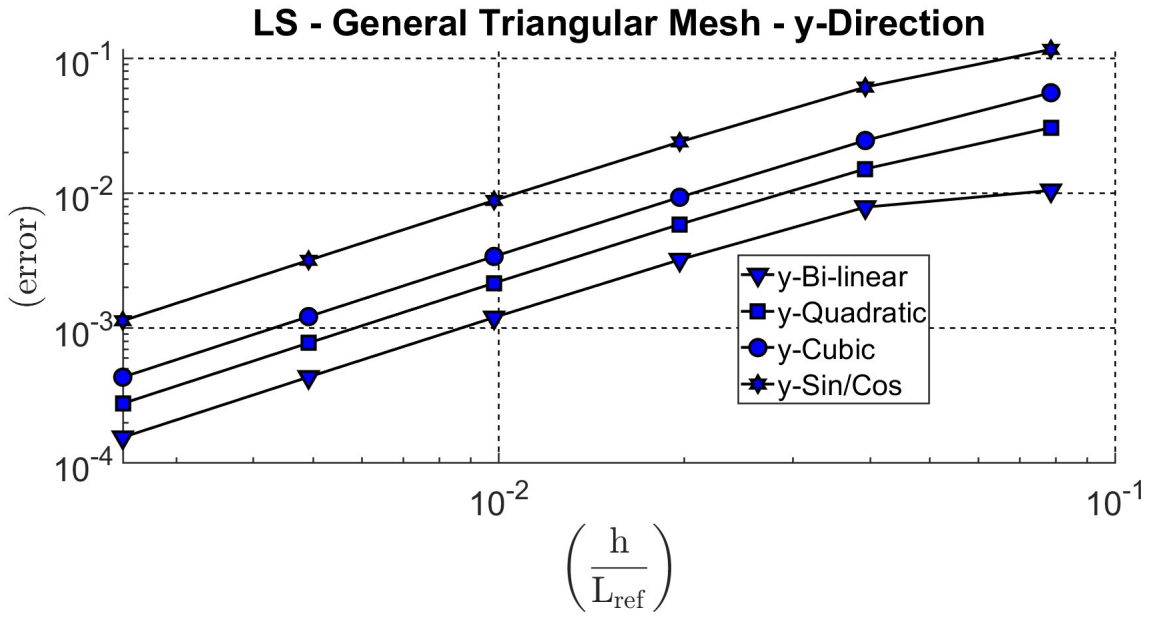


Figure 48: Surface gradient approximation error in the y-direction for the least squares method on a general triangular mesh as a function of the non-dimensional characteristic length.

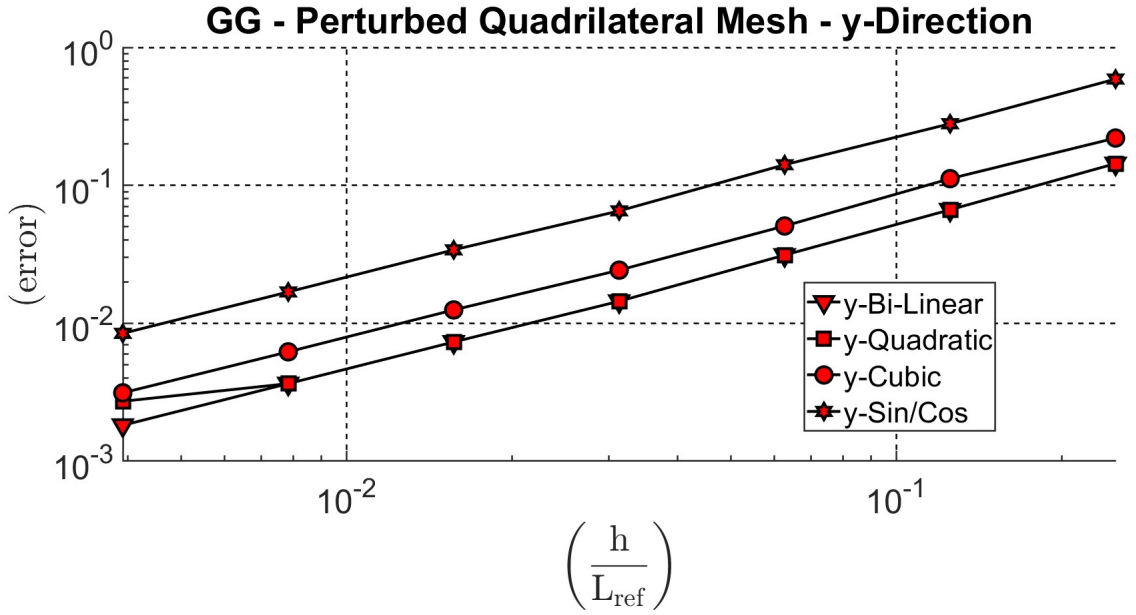


Figure 49: Surface gradient approximation error in the y-direction for the Green-Gauss method on a perturbed quadrilateral mesh as a function of the non-dimensional characteristic length.

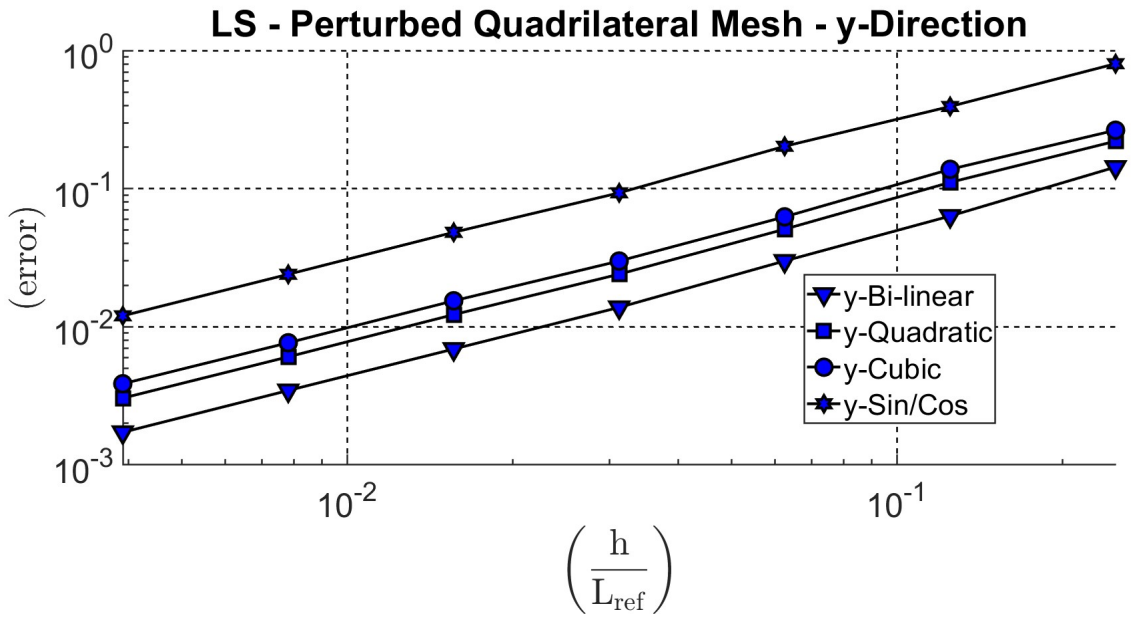


Figure 50: Surface gradient approximation error in the y-direction for the least squares method on a perturbed quadrilateral mesh as a function of the non-dimensional characteristic length.

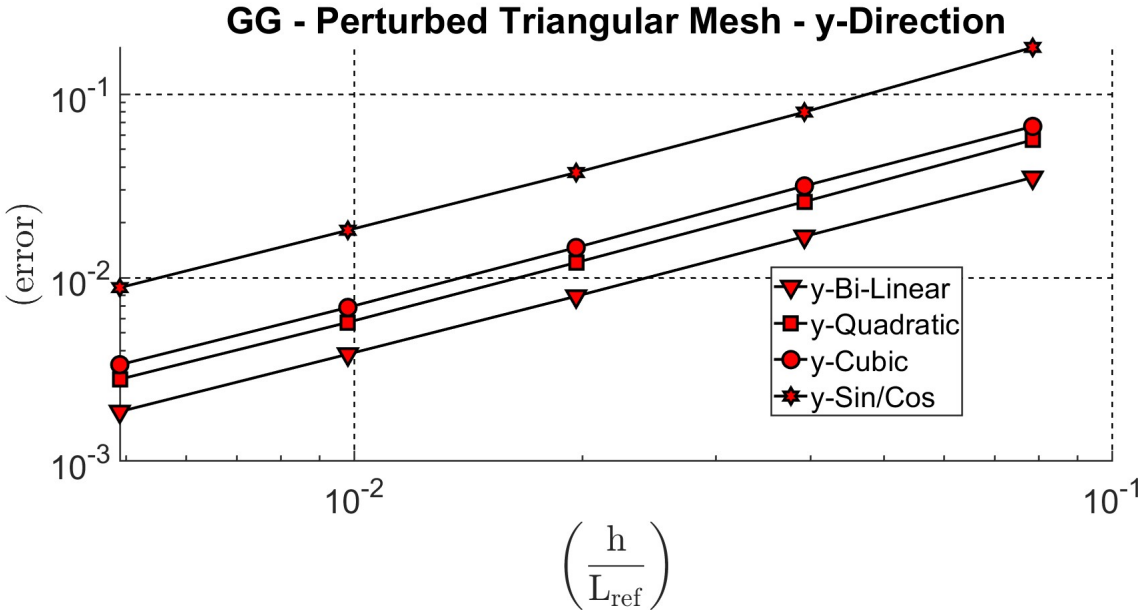


Figure 51: Surface gradient approximation error in the y-direction for the Green-Gauss method on a perturbed triangular mesh as a function of the non-dimensional characteristic length.

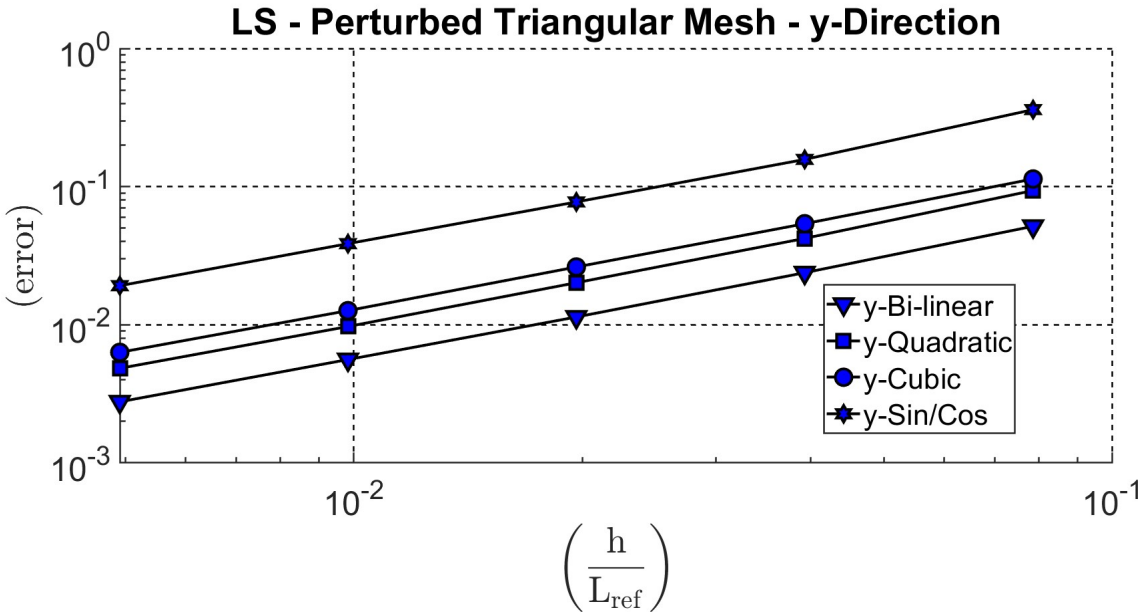


Figure 52: Surface gradient approximation error in the y-direction for the least squares method on a perturbed triangular mesh as a function of the non-dimensional characteristic length.

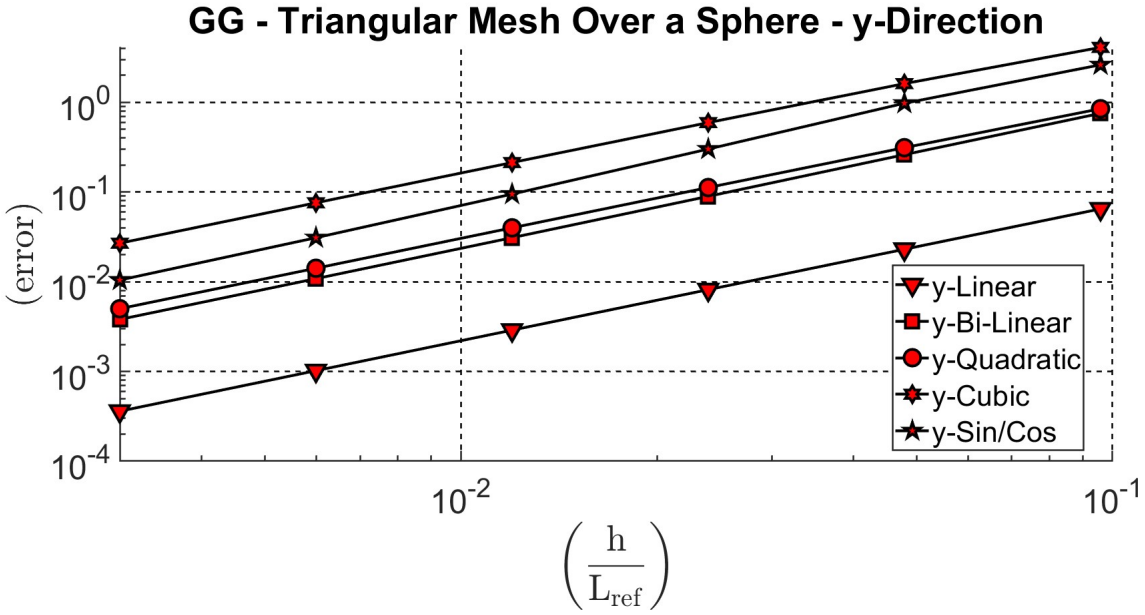


Figure 53: Surface gradient approximation error in the y-directions for the Green-Gauss method on a triangular mesh over the surface of a sphere as a function of the non-dimensional characteristic length.

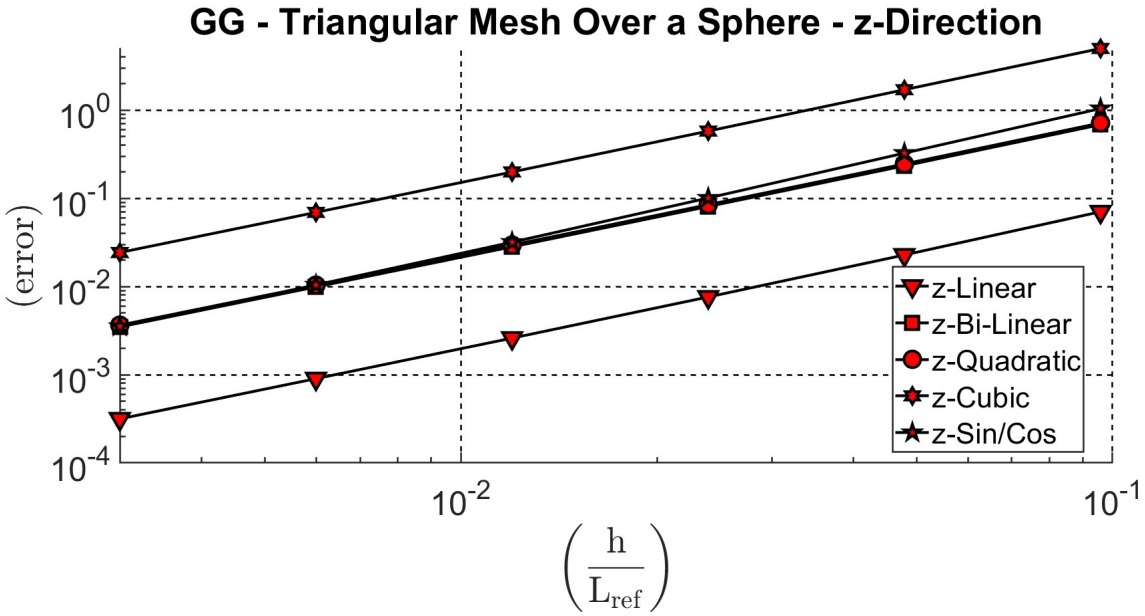


Figure 54: Surface gradient approximation error in the z-directions for the Green-Gauss method on a triangular mesh over the surface of a sphere as a function of the non-dimensional characteristic length.