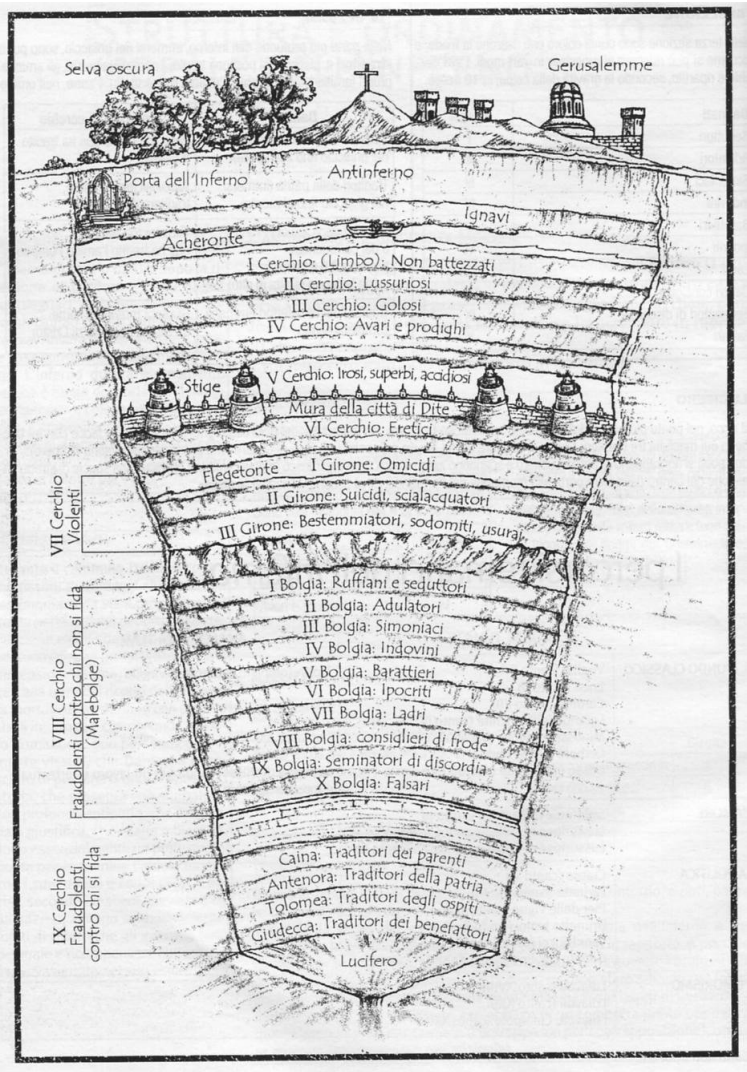
Stability and receptivity of the swept-wing attachment-line boundary layer : a multigrid numerical approach

Gianluca Meneghello

Gianluca Meneghello
gianluca.meneghello@gmail.com

Stability and Receptivity of the Swept-Wing Attachment-Line Boundary Layer: a Multigrid Numerical Approach

(with 51 figures)



Under the supervision of Prof. Peter J. Schmid and Prof. Patrick Huerre
LadHyX - Ecole Polytechnique

Defended on February 15, 2013

On the cover: a multigrid approach to Dante's "Divina Commedia"

A mio padre, per aver sempre trovato il tempo di insegnarmi cose nuove

Nel mezzo del cammin di nostra vita
mi ritrovai per una selva oscura
ché la diritta via era smarrita.

Ahi quanto a dir qual era è cosa dura
esta selva selvaggia e aspra e forte
che nel pensier rinova la paura!

Tant'è amara che poco è più morte;
ma per trattar del ben ch'i' vi trovai,
dirò de l'altre cose ch'i' v'ho scorte.

Dante Alighieri — Inferno, Canto I

Contents

List of Figures

Chapter 1

Introduction and Motivation

The objective of this work is to provide a contribution to the understanding of the broad physical problem concerning the development of perturbations in the flow field around a swept wing.

The swept-wing problem is interesting from both a research and an industrial application perspective.

From a research perspective it is part of the large class of three-dimensional boundary-layer problems, together with rotating disks, rotating cones and rotating spheres [53]. The flow around all these bodies is characterized by a pressure gradient which is not locally aligned with the direction of the velocity vector, as is instead the case, for example, in a pipe flow. As a consequence, the pressure gradient provides an acceleration to only some components of the velocity while leaving others unchanged, thus rotating the velocity vector. Streamlines are accordingly curved in the direction parallel to the gradient, if the flow is accelerated (favorable pressure gradient), and in the direction perpendicular to the gradient, if the flow is decelerated (adverse pressure gradient).

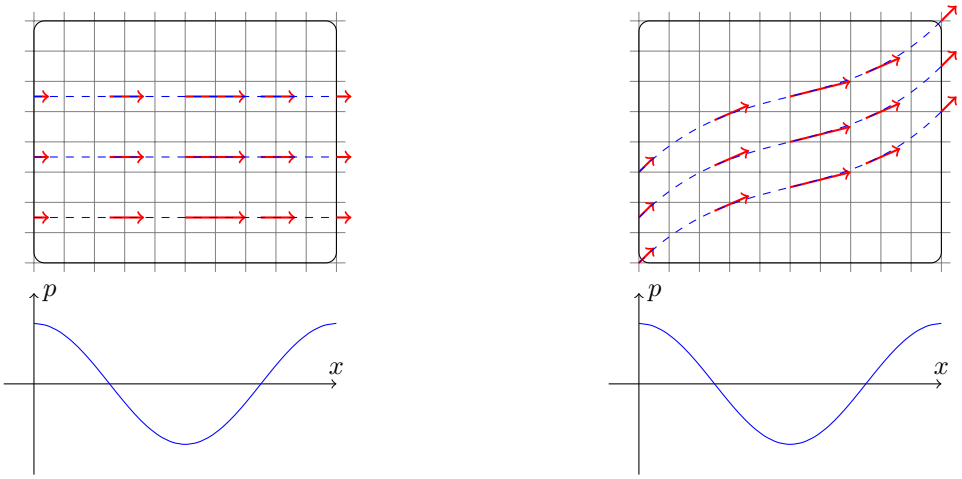


Figure 1.1: The effect of nonalignment of the pressure gradient. On the left: the velocity vector at the inflow is parallel to the pressure gradient, and both are aligned with the horizontal axis. The velocity of the fluid particle moving along the streamline (dashed blue line) increases and decreases, accelerated and decelerated by the pressure gradient, but remains parallel to the horizontal axis. The pressure distribution along the horizontal axis is represented below the plates, as a continuous line. On the right, the velocity vector at the inflow has an angle with the horizontal axis. The horizontal component of the velocity increases and decreases moving along the streamlines, while the vertical component remains unchanged. The velocity vector rotates and the streamlines are curved. The curvature is stronger in the boundary layer close to the plate than in the free stream, because of the lower fluid-momentum there. This is the situation in the boundary layer over a swept wing, if we neglect the curvature of the surface: the pressure gradient is aligned with the chordwise direction, but there is a spanwise velocity component due to the sweep. Streamlines are first curved in the chordwise direction until the pressure minimum is reached, and then curved back into the spanwise direction in the pressure recovery (adverse pressure gradient) region.

This situation is exemplified in Figure 1.1 where we consider two flat plates, viewed from the top, and a pressure distribution such that its gradient is always aligned with the horizontal direction. On the left plate, the flow is aligned with the horizontal direction as well: while moving across the plate from left to right, each fluid particle is first accelerated by a favorable pressure gradient and then decelerated by an adverse pressure gradient, but the velocity vector remains parallel to the horizontal axis, and so do the streamlines. In contrast, on the inflow for the right plate the velocity of our fluid particles has an angle of 45° with the horizontal axis. In this case, the fluid particles moving along the streamline accelerate by increasing their horizontal velocity component as in the previous case, but the vertical velocity component remains unchanged. The velocity vector (hence the streamline) is rotated (curved) towards the horizontal direction. The opposite happens in the presence of an adverse pressure gradient: the horizontal velocity component is reduced while the vertical component remains unchanged. The velocity vectors (and the streamlines) are deflected towards the original flow direction. This effect is stronger in the boundary layer developing at the plate surface than in the free stream, because of the reduced momentum of the fluid there, and the curvature of the streamlines is more pronounced.

At the origin of this nonalignment between the pressure gradient and the velocity vector there can be different phenomena: the centrifugal — or centripetal, depending on the point of view — force in the rotating disk case or the spanwise invariance assumption in the swept-wing case. Whichever the origin, the consequence is the development of strongly three-dimensional boundary layers sharing a characteristic stability behavior: streamwise vorticity is developed within the boundary layer in the form of vortices nearly aligned with the streamlines, known as crossflow vortices. These streamwise structures can be the source of secondary instabilities by lifting low-momentum fluid from the wall towards the free stream and pushing higher-momentum fluid from the free stream towards the wall. Transition from laminar to turbulent fluid motion can be triggered in this manner.

The stability properties of these three-dimensional boundary layers, including the development of associated crossflow vortices and of other types of instabilities, are not yet fully understood, and the contribution of this thesis aims at providing further insights in this direction.

From an industrial point of view, understanding the behavior and origins of disturbances over swept wings is related to our interest in developing passive or active flow control strategies aimed at delaying transition from laminar to turbulent flow to reduce skin-friction drag. Crossflow vortices are deemed principally responsible for this transition for sweep angles greater than $30 - 35°$. Joslin [34, 35] remarks that skin-friction drag amounts to about 50% of the total drag on a subsonic transport aircraft, and that laminar skin friction can be as low as 90% less than turbulent skin friction at the same Reynolds number. Even a partial increase in the extent of the laminar flow can prove very beneficial to the aircraft's efficiency, both by reducing skin-friction drag and by allowing more compact aircraft designs, for example by reducing the amount of fuel that is required to cover a given distance.

Various approaches have been proven effective in increasing the extent of the laminar flow region [34, 35]: among them we mention natural laminar flow, which employs favorable pressure gradients to delay transition, laminar flow control, which employs active blowing and suction through the wing surface, and relaminarization of turbulent flow, again employing suction but at higher energetic cost than laminar flow control. Nonetheless, the effectiveness of natural flow control is limited to small sweep angles (in other words, it is limited to reducing the growth of streamwise Tollmien-Schlichting instabilities, typical of two-dimensional boundary layers; see below).

Despite a large research effort, the cause of transition from laminar to turbulent flow in general and the transition on swept wings in particular still has many unresolved features. The concept of receptivity [47] is recognized to be at the origin of the transition process, and it will play a central role in this

work: disturbances from the environment, e.g., small variations in the incoming free stream, enter the boundary layer and trigger the development of perturbations which may then be amplified and eventually cause transition. Different mechanisms have been proposed in order to describe this amplification and the subsequent transition process. We mention here modal growth, non-normal (transient) growth, the development of secondary instabilities and bypass transition. They are summarized in Figure 1.2, taken from [40].

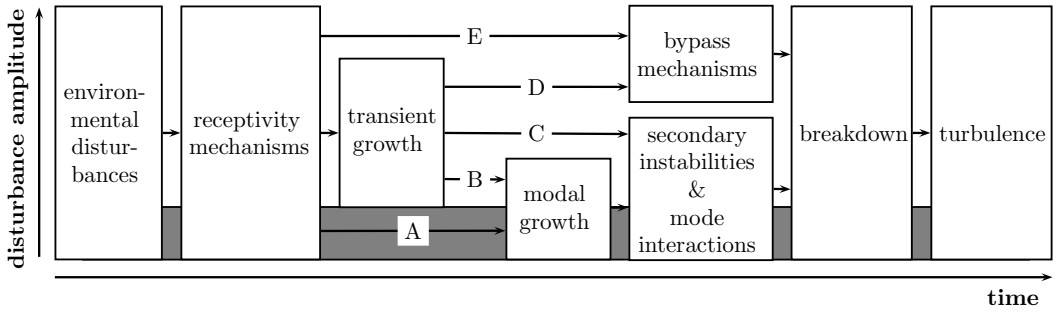


Figure 1.2: Transition mechanisms. Different mechanisms can be responsible for transition from laminar to turbulent flow, but at the origin there is the concept of receptivity: incoming disturbances from the environment — e.g. variations in the incoming free stream — enter the boundary layer and trigger the development of perturbations. These perturbations may then be amplified by different mechanisms, like modal growth or transient growth, may lead to secondary instabilities and/or cause transition to turbulence. A correct understanding of the perturbation behavior and of the receptivity mechanism is essential for increasing the extent of the laminar flow regime in boundary layers by applying control techniques.

Within the modal growth mechanisms only, four possible instabilities have been historically identified that may play a role in the transition process on a swept wing: leading-edge instability and contamination, streamwise (Tollmien–Schlichting) instabilities, centrifugal instabilities and crossflow instabilities [53, 34, 35]. Leading-edge instabilities and contamination are related to the two-dimensional boundary layer developing at the attachment line and can cause the flow to be turbulent over the entire chordwise extent of the wing. Disturbances, stemming from the turbulent boundary layer on the fuselage, can enter the attachment-line boundary layer at the wing root and travel the whole spanwise extent of the wing. Solutions to the contamination problem have been proposed [34, 35] in the form of strong suction at the wing root in order to provide a new, laminar attachment-line boundary layer [51] or by adding a turbulence diverter like the Gaster bump [17]. Streamwise Tollmien-Schlichting-like instabilities are characteristic of the flow in the presence of mildly positive or adverse pressure gradients, i.e. in the vicinity or downstream of the wing section's pressure minimum. They are typical of two-dimensional boundary layers and are the main candidate for causing transition at sweep angles less than 25° [34]. The already mentioned crossflow instabilities coexist with streamwise Tollmien-Schlichting instabilities for sweep angles between 25° and $30 - 35°$ and, for larger sweep angles, overtake Tollmien-Schlichting instabilities in the transition process. Crossflow instabilities are not related to adverse pressure gradients in the same manner as streamwise Tollmien-Schlichting waves are: they can cause transition much closer to the attachment-line, and natural laminar flow designs are not effective in suppressing them.

As suggested by Hall & Seddougui [26] and successively shown by Mack et al. [44], the distinction in attachment-line, crossflow and Tollmien-Schlichting instabilities summarized in Figure 1.3 is to be related to the local approach historically used in the analysis of boundary-layer stability rather than to a real physical difference among the mechanisms. In this local approach, each region of the wing is

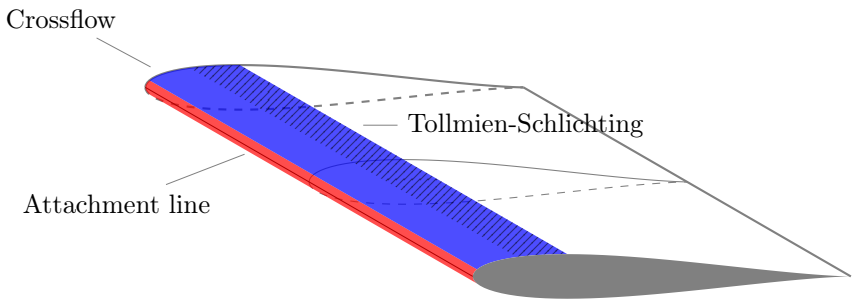


Figure 1.3: Some of the instabilities characterizing the swept attachment-line boundary-layer transition from laminar to turbulent flow and their respective localizations. The attachment-line instabilities mainly occur in the red area, crossflow vortices in the blue area and Tollmien-Schlichting waves in the dashed area. Whether crossflow or Tollmien-Schlichting instabilities dominate the transition to turbulent flow depends on the sweep angle. For sweep angles greater of $30 - 35^\circ$ crossflow instabilities are known to be dominant, and the transition region moves towards the attachment line, thus reducing the extent of the laminar flow regime.

characterized by a simplified flow model — Swept-Hiemenz flow for attachment line, a three-dimensional velocity profile for the crossflow region and a two dimensional boundary layer for the Tollmien-Schlichting region. Stability properties are then analyzed for each simplified flow model. Despite being unable to show the connections among the different regions, the local approach is still useful in providing a first impression of the complexity of the situation as well as an interesting interpretation of the various mechanisms at play.

The Matrix

We have so far identified the interest in and introduced the main characteristics of the swept-wing problem. Driven by a pressure gradient which is not locally aligned with the velocity vectors, the swept-wing flow presents stability characteristics in common with other three-dimensional boundary-layer problems. The development of crossflow vortices is one of these characteristics. The mechanism at the origin of transition from laminar to turbulent flow in three-dimensional boundary layers is not fully understood and, as such, it represents an interesting research subject. At the same time, there is a strong industrial interest in understanding the stability properties of such flows in order to devise and apply passive or active control strategies with the goal of extending the laminar flow regions on aircraft surfaces and, in the end, reduce fuel consumption by reducing drag and allowing the design of more compact aircrafts.

In order to obtain useful insights into the disturbances behavior on a swept wing, we need to choose a representation of the dynamical system underlying it. As a painter can choose among various techniques and points of view to represent the reality surrounding him — oil, tempera, pastel — or a writer can choose between a novel, a poem or a short story, we have to choose a description, or define an abstract model of our dynamical system and select the mathematical and numerical tools we want to apply to it.

The choice of how to represent a dynamical system can be reduced to the choice of an appropriate basis (or space) where this representation is projected onto, and different possibilities are available in this sense. The most natural and obvious, but not necessarily the most useful, is to represent our problem in physical space: in a discrete setting, our basis consists of all degrees of freedom of the problem at each location in space $\mathbf{x}$ and at each time t we wish to represent, and a given state can be represented by a vector in this space. Another approach that can be used in the case of periodic domains, is to choose a Fourier space as a basis, so that wavenumbers replace the positions $\mathbf{x}$ and wave amplitudes replace the

value of the unknown variable at each location $\mathbf{x}$. Owing to the fact that moving from physical space to Fourier space is computationally rather inexpensive, in some cases both representations are used, and selected operations are performed in different spaces.

An alternative choice is to use a modal decomposition: in this case the spatial basis is no longer made up of the single degrees of freedom of the problem but of a composition of them, the eigenvectors of the linearized model (as we will see when presenting multigrid as our numerical method in chapter 4, the eigenvectors can correspond to the Fourier modes in some cases). Eigenvectors represent structures that are invariant in shape — but change in amplitude — during the time evolution of the linearized flow field. The advantage of this representation becomes apparent if we consider that, as has been clearly stated by Mack & Schmid [41], *the goal of any scientific study of a fluid-dynamical process is not in the reproduction of its physical features by direct numerical simulations but in the extraction of the governing underlying mechanisms from the data the DNS produces. In other words, we are interested in the intrinsic flow behavior captured by the dynamics of coherent structures.* The eigenvectors as invariants of the linearized dynamical system can then be interpreted as these coherent structures. This is the representation used in this work to describe the perturbations. The amplitude of each eigenvector, instead of the value of the variable at a location in space, becomes the new degree of freedom, and the dynamics of the system is described by the time evolution of the amplitude associated with each eigenvector.

When all eigenvectors are available, the description of the system provided by the eigenvectors' amplitude is equivalent to the one provided by the original degrees of freedom. Nonetheless, it is often the case that not all the eigenvectors are required (or accessible), and particular features of the flow can be accurately described by a subset of them. For example, when we are interested in the long-time behavior of the perturbations, only the least stable (or the most unstable) eigenvector is of interest, as it is the one that dominates the asymptotic dynamics of the system. More generally, a subset of the eigenvectors can be used to describe the system dynamics, but the choice of which eigenvectors are to be retained or discarded is not obvious *a priori*.

Other choices for a basis are available besides modal decomposition, for example, using a singular value decomposition (SVD) or a proper orthogonal decomposition (POD), each one selecting a different basis for and emphasizing different features of the representation of the space of solutions. Subspaces of these bases can be selected for building reduced-order models of the dynamical system (see for example [55, 60]), but these possibilities are beyond the scope of this work.

The magnifying glass

Once we have selected a representation, or a model, of our problem, we need to focus our attention on a particular characteristic. From the point of view of the perturbations' evolution, the emphasis can be put on their short-time or their long-time behavior. When selecting the features we are more interested in, we must also select the more convenient representation together with a particular numerical approach.

The most natural approach is to observe the system evolving in time, as if it were an experiment. This can be accomplished using a direct numerical simulation (DNS): once an initial condition has been chosen, the governing equations for the perturbations are advanced in time, and the evolution of the flow in physical space can be observed. While this approach can be used for studying both the short-time and the long-time evolution of the flow, it poses two problems: (i) the most obvious one is the dependence of the solution on the initial condition which represents a somewhat arbitrary choice and is of extreme importance for the short-time evolution of the perturbations; (ii) the second one is the difficulty in extracting, from simple observation of the evolution of a particular initial condition, the

coherent structures that represent the true interest of our study (but see [60] for a possible approach).

The short-time behavior is of particular interest for non-normal system and is usually approached in physical space by non-modal analysis [61, 59, 12]. The eigenvalue/eigenvector description, while still possible, is not convenient in analyzing the short-time response because the eigenvectors responsible for the non-normal behavior can be difficult to obtain numerically (but see [49, 50], for a counterexample). The main interest lies in the fact that non-normal systems can present transient growth effects which, as shown in Figure 1.2, can lead to transition even in the case of asymptotically stable flows. The main goal of non-modal analysis is then to investigate the possibility of transient growth and identify the effects of the initial conditions on energy amplification. Of particular interest are the initial conditions leading to the maximum possible transient growth, which are called optimal perturbations: they can be identified by solving an optimization problem for the flow energy (or any equivalent scalar quantity we choose to optimize) over all possible initial conditions. The optimization problem can be treated as a constrained optimization, where the constraints are given by the governing equations and implemented by adjoint fields (also called Lagrangian multipliers). The optimal condition satisfying the constraints is found by identifying stationary points of the Lagrangian. Direct and adjoint equations are obtained by this procedure and can be marched forward and backward in time in order to determine the optimal initial condition. In this sense, non-modal analysis removes the arbitrariness of the initial condition by identifying the most interesting one.

In contrast, the eigenvalue/eigenvector description is more often used in the description of the long-time (asymptotic, or modal) behavior of the system. As we will see in detail, this description is used in this work to identify the coherent structures underlying the flow as well as to identify how the behavior of these structures is influenced by external disturbances. A Lagrangian approach similar to the one just described is used to identify receptivity and sensitivity of an eigenvector (and its corresponding eigenvalue) to forcing and structural modifications of the operator. The adjoint equations and their solutions given by the adjoint fields will play a central role in this process.

Stato dell'arte

The swept-wing attachment-line boundary layer has been investigated from multiple points of view in the past. References to most of the relevant work dating before 2003 can be found in the reviews by Reed and Saric [53, 54, 57, 58].

Most of the research has been concentrated on local models: the swept Hiemenz flow, characterizing the flow impinging on a flat plate with a sweep angle, has been extensively studied as a local approximation of the flow close to the attachment line of a swept wing. DNS computations have been performed by Joslin [32, 33] while the short-time optimal growth has been the subject of the work of Guégan et al. [23, 22, 24] and Obrist & Schmid [49]. Comparison of theoretical and experimental results on crossflow instabilities have been provided by Dagenhart & Saric [13].

In this work, the global modal approach for a realistic configuration of the swept wing is attempted. The same approach has been applied to the swept Hiemenz flow starting from the work of Hall, Malik & Poll [25] who studied the stability of the Görtler-Hämmerlin mode (i.e. a mode having the same streamwise structure as the swept Hiemenz flow) and showed that this three-dimensional flow can, in contrast to the two-dimensional (unswept) Hiemenz flow, become unstable above a critical Reynolds number.

Lin & Malik [38] extended the work of Hall, Malik & Poll by computing several modes of the incompressible swept Hiemenz flow using a Chebyshev spectral collocation method and regular polynomials

$\{P(x) = x^n, n = 0, 1, 2, \dots\}$ in order to discretize the normal- and chord-wise direction of their domain. They identified a branch of eigenvalues, all moving at approximately the same phase speed of $c_r = 0.35$ in the spanwise direction — i.e. moving at a velocity equal to 0.35 times the spanwise free-stream velocity component. It was shown that the most unstable mode was the symmetric Görtler-Hämmerlin mode already found by Hall et al. [25]. Less unstable modes were shown to alternate between antisymmetric and symmetric as one descends down the branch towards smaller growth rates.

In a subsequent study [39] they addressed the question of the leading-edge curvature by using a second-order boundary-layer approximation in computing the base flow and showed that the flow was stabilized by increasing the leading-edge curvature. For small distances in the chord-wise direction from the attachment line, this effect was mainly related to the curvature terms appearing in the continuity equation, while the centrifugal acceleration terms in the momentum equations have been found of less importance.

With a similar approach, Obrist & Schmid [48] addressed the same problem by replacing the regular polynomials used by Lin & Malik in the chord-wide discretization with Hermite polynomials. They too found the most unstable mode to be the Görtler-Hämmerlin mode, showing an exponential decay outside the boundary layer, and identified a richer spectrum composed of several branches, continuous and discrete. The non-modal analysis previously mentioned [49] was part of the same work.

Almost all the investigations mentioned so far addressed the simplified model of swept Hiemenz flow. The first global analysis of the leading-edge region was performed by Mack, Schmid & Sesterhenn [44] and Mack & Schmid [42, 43]. They addressed the stability of a compressible flow impinging on a parabolic body with a sweep angle. A high-order finite-difference discretization was used in both the normal and chordwise direction. They showed a global spectrum consisting of different branches: boundary layer modes, acoustic modes and wave-packet modes. Of these, the boundary layer and wave-packet branches are of interest for the current, incompressible study. Additionally, they showed, for the first time, evidence of a connection between attachment-line and crossflow instabilities, a feature already suggested but never proven in previous works [26]. This result was made possible by considering a domain extending beyond the attachment-line region.

Outline

The present work continues in the wake of the global modal approach used by Mack et al. [40], but analyzes an incompressible flow instead of a compressible one and extends Mack's results by including a receptivity and sensitivity analysis. This work is organized in six chapters, the first being this introduction.

In chapter 2 we define the swept-wing problem: governing equations for the base flow and the perturbations are derived, and the dimensionless parameters governing the base flow and perturbation behavior will be described. The adjoint equations for the perturbations are obtained starting from a Lagrangian description of the flow and, building on that, receptivity to external forcing and sensitivity to structural modifications will be defined. The concept of the wavemaker in the context of global analysis, as introduced by Giannetti & Luchini [19], will be presented.

Once the theoretical framework underlying the swept-wing flow analysis is presented, our numerical approach is specified. Because numerical issues represent a large part of this work, it is split into two parts. The first part, in chapter 3, includes details related to the discrete representation of our problem: the pressure-form of the nonlinear and linear Navier-Stokes equations — where a Poisson equation for the pressure replaces the mass-conservation equation — and their discretization, the boundary conditions,

the grid generation process and the validation of the implemented discretization. A final section contains some considerations on the required grid size. The second part, in chapter 4, is dedicated to the solver used to compute the base flow, based on the multigrid framework. An analysis of the multigrid algorithm from the point of view of an iterative solution process is performed, and the most important parts of the algorithm are analyzed using Local Fourier Analysis (LFA), a useful tool providing theoretical estimates on the convergence rates of iterative methods. The two available algorithmic choices are then introduced, starting from the more well-known Correction Scheme and moving on to the less well-known but more powerful Full Approximation Scheme. At the end of the chapter, some simple test cases are used to identify and overcome possible problems leading to a loss of efficiency.

In chapter 5 we return to the theory covered in chapter 2 and present the results obtained for the swept-wing configuration. A description of the main features of the base flow is followed by a presentation of the results obtained by a global analysis of the linearized Navier-Stokes operator, including the computed part of the spectrum and an in-depth analysis of the least stable eigenvector. The corresponding adjoint field is then analyzed, and the significance of its distribution in the domain is clarified. The identification of the wavemaker region concludes this chapter.

Finally, chapter 6 summarizes the main results of this work and describe some paths that can be followed to build upon what has been accomplished here.

A short Matlab code is provided in Appendix A This is a demonstration code used in producing the results for the test cases at the end of the chapter on multigrid (in section 4.6), but it is not the code used in computing the main results of this work presented in chapter 5. It can nonetheless be useful in providing an outline of how a real multigrid code can be structured.

I found Figure 1.4, from a lecture given by Peter Schmid at Institut Henri Poincaré in Paris, very useful in outlining the main steps required in a receptivity and sensitivity analysis. While the chapters in this work do not follow exactly the same outline, this flowchart can be helpful in maintaining a broad view on this work and in avoiding getting lost in details. The unfilled boxes on the right briefly indicate the numerical tools used at each step.

As a final note in this introduction, I would like to mention that in writing these pages I tried as much as possible to follow and apply the suggestions given by McIntyre in his paper *Lucidity and science I: Writing skills and the pattern perception hypothesis* [46]. Whether or not I accomplished this task is another story, but McIntyre's paper is nonetheless a very interesting reading that deserves to be mentioned.

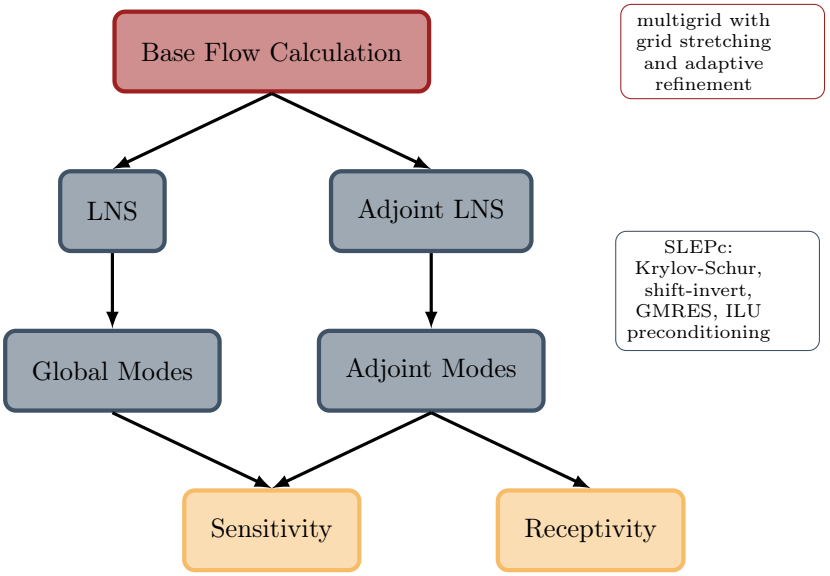


Figure 1.4: Receptivity and sensitivity analysis flow chart (courtesy of P.J. Schmid). The first step of the analysis is the computation of a base flow, the stability properties of which we are interested in. The governing equations for the perturbations can then be derived by linearizing the Navier-Stokes equations around this base flow and are named the linearized Navier-Stokes equations (LNS). Similarly, their adjoint counterpart can be obtained. Solution of the direct and adjoint eigenvalue problems returns the direct and the adjoint modes (which can also be seen as the right and left eigenvector of the direct problem). Receptivity analysis involves only the adjoint modes, while sensitivity analysis involves both the direct and adjoint modes.

Chapter 2

Problem Definition

The theoretical framework underlying our problem is presented in this chapter. The geometry of interest is the leading-edge region of a swept wing and is described first. The equations governing incompressible, viscous flow are then introduced by defining the nonlinear Navier-Stokes operator $\mathcal{R}(\mathbf{q})$, where $\mathbf{q}$ is the state vector consisting of the three components of the velocity field $\{u, v, w\}$ as well as the pressure p. The reference scales for length, velocity and time are introduced, which make $\mathcal{R}(\mathbf{q})$ dimensionless. This also yields dimensionless parameters governing the fluid motion — namely, the chord-based Reynolds number Re_C and the sweep angle Λ.

Our main interest lies in the evolution of small perturbations superimposed on a given stationary flow, the receptivity of these perturbation to external forcing and their sensitivity to structural changes in the governing equations. Receptivity and sensitivity form the foundation for the passive and active manipulation of the flow by applying control-theoretic means.

The equations governing the evolution of infinitesimal perturbations are derived based on a linearization of the Navier-Stokes operator $\mathcal{R}$ around a stationary flow $\mathbf{Q}$ with $\mathcal{R}(\mathbf{Q}) = 0$. The linearized Navier-Stokes equations can then be represented by the application of a linear operator $\mathcal{L}$ to a perturbation field $\delta\mathbf{q}$, i.e. $\mathcal{L}\,\delta\mathbf{q} = \left(\partial\mathcal{R}/\partial\mathbf{q}\right)\delta\mathbf{q}$.

The linear adjoint operator $\mathcal{L}^+$, which plays a central role in receptivity and sensitivity problems, is then derived by defining a Lagrangian $\mathcal{I}$ for a generic objective functional obj. The governing equations, boundary and initial conditions are implemented using adjoint fields (also known as Lagrange multipliers). The search for a stationary point of the Lagrangian $\mathcal{I}$ as a point in state space representing a solution of the dynamical problem corresponds to setting to zero the gradients (also known as variations) of the Lagrangian with respect to all variables: the particular case of setting to zero the gradient with respect to the direct variables $\mathbf{q}$ provides the adjoint equations.

We then consider the possibility of forcing the linearized Navier-Stokes equations by a generic force $\mathbf{f}'$ such that $\mathcal{L}\delta\mathbf{q} = \mathbf{f}'$ and proceed to assess the effects of this forcing. Receptivity and sensitivity analyses address this configuration for the case of a small variation in the forcing. In order to show this quantitatively, the Lagrangian functional $\mathcal{I}$ is redefined as a function of the state variables given by the flow field, the perturbation field and the unknown forcing field. The gradient of the Lagrangian $\mathcal{I}$ with respect to the forcing $\mathbf{f}'$ will provide us with a measure of receptivity of the objective functional to the forcing.

As a last step, we consider the effect of a structural change in the governing equations on the spectrum. The Lagrangian functional $\mathcal{I}$ is redefined again in order to take into account the operator representing the governing equations and an eigenvalue as variables, and its variations are computed with respect to the new variables, resulting in an equation for the change in a particular eigenvalue as a function of a change in the operator.

2.1 The governing equations for the base flow

The flow in the leading-edge region of an infinitely long swept wing is considered. The wing, as represented in Figure 2.1, is subjected to a uniform flow $\mathbf{U}^*_\infty = \{U^*_\infty, 0, W^*_\infty\}$, where U^*_∞ and W^*_∞ are the chordwise and spanwise components of the uniform velocity field. The sweep angle Λ then satisfies the relation $\tan\Lambda = W^*_\infty/U^*_\infty$. Asterisks denote dimensional quantities.

Two reference systems are defined as shown in Figure 2.1, both centered at the leading edge: (i) a Cartesian coordinate system, whose x-axis is parallel to the chord, whose y-axis points upwards and whose z-axis is in the spanwise direction and (ii) a curvilinear coordinate system, with its s-axis running along the profile, its n-axis normal to the profile surface and sharing its z-axis with the Cartesian coordinate system.

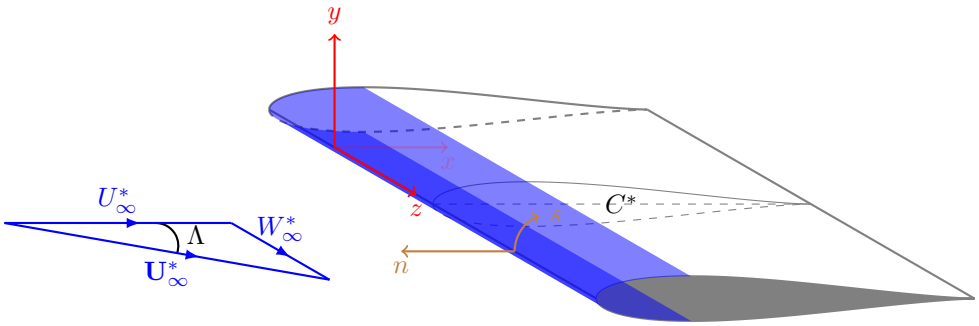


Figure 2.1: Sketch of the geometry. The Cartesian reference system is displayed in red and the curvilinear one in brown. The velocity components of the free-stream, uniform flow are shown in blue. A zero angle of attack will be considered, i.e., V_∞ will be set to zero throughout this work.

The flow is governed by the incompressible Navier-Stokes equations ($\mathcal{R}$), where a forcing term $\mathbf{f}$ is accounted for explicitly.

$$\mathcal{R}(\mathbf{q}) \equiv \begin{cases} \partial_t \mathbf{u} + \nabla \mathbf{u}\,\mathbf{u} - \nu\Delta\mathbf{u} + \nabla p &= \mathbf{f} \\ \nabla\cdot\mathbf{u} &= 0 \end{cases} \qquad (\mathcal{R})$$

Δ denotes the Laplacian operator, and ν is the dimensionless viscosity taken as the inverse of the Reynolds number. The quantity $\mathbf{q} = \{\mathbf{u}, p\} = \{u, v, w, p\}$ is a vector containing all state variables.

The Navier-Stokes equations ($\mathcal{R}$) are to be complemented by boundary conditions on the inflow Γ_{in}, outflow Γ_{out} and on the solid boundary Γ_{sb}, as well as by initial conditions at $t = 0$.

As boundary conditions, a Dirichlet-type condition is specified on both the inflow and the solid boundary for all components of velocity. No boundary condition is required for the pressure in this formulation, but a global constraint $\int_\Omega p\, d\Omega = c$ has to be specified, where c is an arbitrary constant which can be set to zero without loss of generality. The situation will change in the pressure-equation formulation that will be introduced in the next chapter, where a boundary condition for the pressure will be required. At this stage, we neglect the global constraint on the pressure as well as the boundary conditions for velocity at the outflow — they will be clarified in the next chapter. We can nonetheless specify a generic boundary condition by means of applying an operator $\mathcal{H}_\mathcal{R}$ to the vector $\mathbf{q}$. For the

boundary conditions specified so far we can write

$$\mathcal{H}_{\mathcal{R}}(\mathbf{q}) \equiv \begin{cases} \mathbf{u}(\mathbf{x},t) - \mathbf{u}_{\Gamma}(\mathbf{x}) = 0 & \text{for } \mathbf{x} \text{ on } \Gamma_{in}, \\ \mathbf{u}(\mathbf{x},t) = 0 & \text{for } \mathbf{x} \text{ on } \Gamma_{sb}. \end{cases} \tag{2.1}$$

In a similar way, initial conditions on velocities can be specified by means of applying an operator $\mathcal{G}_{\mathcal{R}}$ to the vector $\mathbf{q}$ as

$$\mathcal{G}_{\mathcal{R}}(\mathbf{q}) \equiv \quad \mathbf{u}(\mathbf{x},0) - \mathbf{u_0}(\mathbf{x}) = 0. \tag{2.2}$$

No initial condition is required for the pressure.

All quantities in the previous equations — ($\mathcal{R}$), (2.1) and (2.2) — are rendered dimensionless by choosing the chord C^*, defined perpendicular to the leading edge, as the reference length, the chordwise velocity component U^*_∞ as the reference velocity and $T^* = C^*/U^*_\infty$ as the reference time scale. The reference pressure is taken as $\rho^* U^{*2}_\infty$, where ρ^* is the dimensional density of the fluid. A chord-based Reynolds number can be defined as

$$Re_C = \frac{U^*_\infty C^*}{\nu^*} \tag{2.3}$$

where ν^* is the dimensional kinematic viscosity of the fluid. The dimensionless viscosity ν in the governing equations ($\mathcal{R}$) is then taken as the inverse of Re_C according to

$$\nu = \frac{1}{Re_C} = \frac{\nu^*}{U^*_\infty C^*}.$$

Dimensional quantities can be recovered from solutions of $\mathcal{R}(\mathbf{q})$ as

$$u^* = U^*_\infty u, \qquad v^* = U^*_\infty v, \qquad w^* = U^*_\infty w, \qquad p^* = \rho^* U^{*^2}_\infty p$$

while dimensional time and lengths can be obtained by multiplying their dimensionless values by T^* and C^*, respectively.

Once the equations are made dimensionless and for a given geometry, two governing parameters are sufficient to completely define the problem: the Reynolds number Re_C and the sweep angle Λ. The dimensionless spanwise component of the uniform flow is obtained as $W_\infty = \tan\Lambda$.

Other dimensionless parameters

An alternative set of dimensionless parameters can be used in the context of swept-wing boundary-layer analysis: (i) the Reynolds number Re_r, based on the leading-edge radius r^* and the chordwise velocity U^*_∞, and (ii) the viscous Reynolds number Re_s, based on the viscous length δ^* and the spanwise velocity W^*_∞. We have

$$Re_r = \frac{U^*_\infty r^*}{\nu^*}, \tag{2.4}$$

$$Re_s = \frac{W^*_\infty \delta^*}{\nu^*}. \tag{2.5}$$

The viscous length scale δ^* is defined as

$$\delta^* = \sqrt{\frac{\nu^*}{S^*}} \tag{2.6}$$

where $S^* = 2U^*_\infty/r^*$ is the strain rate at the attachment line for the inviscid flow around a cylinder with radius equal to the leading-edge radius. This somewhat arbitrary choice for the strain rate S^* is necessary since the actual strain rate is not known in advance, and reflects the use of the asymptotic strain rate in the closely related problem of swept Hiemenz flow [24, 48]. Substituting (2.6) into (2.5) and using $\tan\Lambda = W^*_\infty/U^*_\infty$ it can be shown that the following relation holds:

$$Re_s = \sqrt{Re_r}\tan\Lambda \tag{2.7}$$

such that specifying the two Reynolds numbers determines the sweep angle Λ. The relation between the chord-based and the radius-based Reynolds number is

$$Re_r = \frac{r^*}{C^*}Re_C = rRe_C$$

where r is the dimensionless leading-edge radius.

Base flow computation

Obtaining a steady state solution of the Navier-Stokes equations ($\mathcal{R}$) is a necessary step for studying the stability of the perturbations. A possible exception is the stability analysis of a periodic, oscillatory base flow by means of Floquet analysis, but this option will not be considered in this work.

For the case of a stable, steady state solution, two options are available from the computational point of view: the first is to advance the system in time up until convergence to a flow state which remains invariant in time is reached; the second is to directly tackle the steady-state equations with a Newton or a Newton-like method. A Newton-like method is used in this work and will be introduced in chapter 4.

It is worth noting that a stable solution is not always available. In fact, it is often the case that the properties of an unstable configuration are analyzed in order to understand the mechanisms underlying the instability. In this case, direct time stepping is not an option as the computed field will diverge from the one of interest towards another stable configuration. A possibility is then to employ a Newton method and proceed by continuation.

2.2 The governing equations for the perturbations

We now suppose that a steady-state flow, represented by the state vector $\mathbf{Q} = \{U, V, W, P\}$, has been obtained by solving the Navier-Stokes equation $\mathcal{R}(\mathbf{Q}) = 0$ or their steady state equivalent. We can then investigate the evolution of a small perturbation $\delta\mathbf{q} = \{\delta u, \delta v, \delta w, \delta p\}$ superimposed on the base state $\mathbf{Q}$, i.e. the evolution of $\mathbf{q} = \mathbf{Q} + \delta\mathbf{q}$.

The perturbation field is governed by the linearized Navier-Stokes equations $\mathcal{L}$ which can be obtained by computing the Jacobian of the Navier-Stokes operator $\mathcal{R}$, evaluated at the given flow state $\mathbf{Q}$,

$$\mathcal{L}\,\delta\mathbf{q} = \left.\frac{\partial\mathcal{R}}{\partial\mathbf{q}}\right|_{\mathbf{Q}}\delta\mathbf{q}. \tag{2.8}$$

In order to simplify notation for the remainder of this chapter, we omit the prefix δ for the perturbation variables. The new perturbation variable is $\mathbf{q} = \{u, v, w, p\}$, while a capital $\mathbf{Q}$ will be used to indicate the solution of the nonlinear equations.

The linearized equations read

$$\mathcal{L}\,\delta\mathbf{q} = \left(\begin{bmatrix} \partial_t & & & \\ & \partial_t & & \\ & & \partial_t & \\ & & & 0 \end{bmatrix} + \begin{bmatrix} \bar{\mathcal{Q}}_\nu + U_x & U_y & U_z & \partial_x \\ V_x & \bar{\mathcal{Q}}_\nu + V_y & V_z & \partial_y \\ W_x & W_y & \bar{\mathcal{Q}}_\nu + W_z & \partial_z \\ \partial_x & \partial_y & \partial_z & 0 \end{bmatrix} \right) \begin{bmatrix} u \\ v \\ w \\ p \end{bmatrix} = \begin{bmatrix} f'_u \\ f'_v \\ f'_w \\ 0 \end{bmatrix} \qquad (\mathcal{L})$$

where

$$\bar{\mathcal{Q}}_\nu = U\partial_x + V\partial_y + W\partial_z - \nu\left(\partial_{xx} + \partial_{yy} + \partial_{zz}\right) \tag{2.9}$$

is the convection-diffusion operator and the subscripts x, y, z represent partial derivatives with respect to the corresponding coordinates. The time derivative has been kept separate to explicitly state that the problem is a differential-algebraic system: no time derivative appears for the pressure. A forcing term $\mathbf{f}'$ is introduced explicitly. The effect of the $\mathcal{L}$ operator — and in particular of the base flow velocity $\mathbf{U}$ and its gradient — on the perturbation velocity field can be split in two parts: (i) a transport-diffusion mechanism, summarized by the operator $\mathcal{Q}_\nu$, and (ii) a production mechanism, represented by the gradients of the base flow [45].

Appropriate boundary and initial conditions are to be defined for the perturbation equations. As a boundary condition, we force the perturbation velocity to zero at the inflow and at the solid boundary (outflow boundary condition and pressure will be dealt with in the next chapter), i.e.,

$$\mathcal{H}_\mathcal{L}\left(\mathbf{q}\right) \equiv \qquad \mathbf{u}\left(\mathbf{x}, t\right) = 0 \qquad \text{for } \mathbf{x} \text{ on } \Gamma, \tag{2.10}$$

and a given initial condition is imposed as

$$\mathcal{G}_\mathcal{L}\left(\mathbf{q}\right) \equiv \qquad \mathbf{u}\left(\mathbf{x}, 0\right) - \mathbf{u_0}\left(\mathbf{x}\right) = 0. \tag{2.11}$$

Because all coefficients in the $\mathcal{L}$ operator are constant in time, a Laplace transform of the perturbation variables can be performed and solutions to $(\mathcal{L})$ can be sought as a linear combination of global modes, each having the form

$$(u, v, w, p) = (\hat{u}, \hat{v}, \hat{w}, \hat{p})\, e^{\sigma t} \tag{2.12}$$

where σ is a complex-valued number and $\hat{u}, \hat{v}, \hat{w}, \hat{p}$ are complex-valued fields which depend on σ but not on time. As a consequence, the $\mathcal{L}$ operator is transformed into the $\hat{\mathcal{L}}$ and the linearized Navier-Stokes system $(\mathcal{L})$ reads

$$\hat{\mathcal{L}}\,\hat{\mathbf{q}} = \left(\sigma \begin{bmatrix} 1 & & & \\ & 1 & & \\ & & 1 & \\ & & & 0 \end{bmatrix} + \begin{bmatrix} \bar{\mathcal{Q}}_\nu + U_x & U_y & U_z & \partial_x \\ V_x & \bar{\mathcal{Q}}_\nu + V_y & V_z & \partial_y \\ W_x & W_y & \bar{\mathcal{Q}}_\nu + W_z & \partial_z \\ \partial_x & \partial_y & \partial_z & 0 \end{bmatrix} \right) \begin{bmatrix} \hat{u} \\ \hat{v} \\ \hat{w} \\ \hat{p} \end{bmatrix} = \begin{bmatrix} \hat{f}'_u \\ \hat{f}'_v \\ \hat{f}'_w \\ 0 \end{bmatrix} + \begin{bmatrix} u_0 \\ v_0 \\ w_0 \\ 0 \end{bmatrix} \tag{2.13}$$

where the right-hand side contains the Laplace transform of the right-hand side of $(\mathcal{L})$ and the initial conditions for the velocity field (u_0, v_0, w_0).

The homogeneous equivalent of (2.13), i.e. when the right-hand side forcing term $\mathbf{f}'$ and the initial condition $\mathbf{u_0}$ are zero, can be recast as a generalized eigenvalue problem in the form

$$\left(\sigma B + A\right)\hat{\mathbf{q}} = 0 \tag{2.14}$$

where B and A are the first and the second matrix in equation (2.13) — or an appropriate discretization.

2.3 The adjoint governing equations for the perturbations

The adjoint equations are a key tool in understanding the effects of the forcing term $\mathbf{f}'$ on the perturbations $\mathbf{q}$ — or, as we will see later, of $\hat{\mathbf{f}}'$ on $\hat{\mathbf{q}}$. Adjoint equations can be obtained by starting from an optimization problem: an objective functional, for example the energy of the perturbations $\int_\Omega \mathbf{u}\cdot\mathbf{u}\, d\Omega$ in the domain Ω, is maximized under the constraints given by the perturbations' governing equations ($\mathcal{L}$) and corresponding boundary conditions. The constraints can be implemented directly into the Lagrangian by means of multipliers (also known as the adjoint fields)

$$\mathcal{I}\left(\mathbf{q},\mathbf{q}^+,\mathbf{q}_0,\mathbf{q}_0^+,\mathbf{q}_\Gamma,\mathbf{q}_\Gamma^+\right) = obj - \left\langle \mathbf{q}^+, \mathcal{L}\left(\mathbf{Q}\right)\mathbf{q} - \mathbf{f}'\right\rangle_\Omega^T - \left\langle \mathbf{q}_0^+, \mathcal{G}_\mathcal{L}\mathbf{q}_0\right\rangle_\Omega - \left\langle \mathbf{q}_\Gamma^+, \mathcal{H}_\mathcal{L}\mathbf{q}_\Gamma\right\rangle_{\partial\Omega}^T \tag{2.15}$$

where the fact that the perturbation equation $\mathcal{L}\left(\mathbf{Q}\right)$ and associated initial $\mathcal{G}_\mathcal{L}$ and boundary $\mathcal{H}_\mathcal{L}$ conditions are linear is made clear by using a matrix-vector product notation. The objective functional obj is left unspecified for the time being.

The inner products are defined as integrals over time and the domain Ω for the perturbation equations, over the domain Ω for the initial condition and over time and the boundary $\delta\Omega$ for the boundary conditions:

$$\left\langle \mathbf{a},\mathbf{b}\right\rangle_\Omega^T = \int_0^T\int_\Omega \mathbf{a}^H\,\mathbf{b}\, d\Omega\, dt \qquad \left\langle \mathbf{a},\mathbf{b}\right\rangle_\Omega = \int_\Omega \mathbf{a}^H\,\mathbf{b}\, d\Omega \qquad \left\langle \mathbf{a},\mathbf{b}\right\rangle_{\partial\Omega}^T = \int_0^T\int_{\partial\Omega} \mathbf{a}^H\,\mathbf{b}\, ds\, dt \tag{2.16}$$

where ds represents a differential element of the domain boundary $\partial\Omega$.

Once the Lagrangian functional is defined, the optimization problem is solved by searching for stationary points of the Lagrangian, which can be easily identified by setting its gradients (also called variations) with respect to all independent variables — direct and adjoint — to zero. The gradient with respect to a given variable $\mathbf{a}$ is defined as

$$\frac{\partial\mathcal{I}}{\partial\mathbf{a}}\delta\mathbf{a} = \lim_{s\to 0}\frac{\mathcal{I}\left(\mathbf{a}+s\delta\mathbf{a}\right)-\mathcal{I}\left(\mathbf{a}\right)}{s} \tag{2.17}$$

which, in effect, recovers the same functional derivative used in the derivation of the governing equations for the perturbations ($\mathcal{L}$). The variable $\mathbf{a}$ can take the form of a scalar, a vector or a matrix.

Recalling that all operators involved in the inner products, as well as the inner products themselves, are linear, the gradients can be readily obtained. Evaluation of the gradients with respect to the adjoint variables $\mathbf{q}^+,\mathbf{q}_0^+,\mathbf{q}_\Gamma^+$ are immediate and recover the governing equations ($\mathcal{L}$), boundary conditions (2.10) and initial condition (2.11) operators. For example, the gradient with respect to $\mathbf{q}^+$ reads

$$\frac{\partial\mathcal{I}}{\partial\mathbf{q}^+}\delta\mathbf{q}^+ = \frac{\int_0^T\int_\Omega\left(\mathbf{q}^+ + s\delta\mathbf{q}^+\right)\left(\mathcal{L}\mathbf{q}-\mathbf{f}\right) - \mathbf{q}^+\left(\mathcal{L}\mathbf{q}-\mathbf{f}\right)d\Omega\, dt}{s} = \int_0^T\int_\Omega \delta\mathbf{q}^+\left(\mathcal{L}\mathbf{q}-\mathbf{f}\right)d\Omega\, dt = 0.$$

Requiring the equation to be satisfied for any $\delta\mathbf{q}^+$, any control volume Ω and any integration time T is equivalent to imposing the perturbation equations $\mathcal{L}\mathbf{q}=\mathbf{f}$.

Computing gradients with respect to the direct variables is less straightforward, as these variables do not appear explicitly in the inner product. Using integration by parts, the direct variables can be made explicit and the gradients can be computed using the same approach as for the adjoint. The operation can be performed term-by-term for the operator $\mathcal{L}$. For example, integration by parts is applied to the

operator $\partial_t + \bar{\mathcal{Q}}_\nu$ (2.9), which is the equivalent of a scalar convection-diffusion equation, to extract the scalar velocity component u. We obtain

$$\left\langle u^+, \left[\partial_t + \bar{\mathcal{Q}}_\nu\right] u\right\rangle_\Omega^T = \left\langle \left[-\partial_t + \bar{\mathcal{Q}}_\nu^+\right] u^+, u\right\rangle_\Omega^T + \mathcal{J} \tag{2.18}$$

where the linear adjoint convection-diffusion operator $\bar{\mathcal{Q}}_\nu^+$ operating on the adjoint field reads

$$\bar{\mathcal{Q}}_\nu^+ = -U\partial_x - V\partial_y - W\partial_z - \nu\left(\partial_{xx} + \partial_{yy} + \partial_{zz}\right). \tag{2.19}$$

Looking at the result of the integration by parts of the convection-diffusion equation (2.18) and at the adjoint operator $\bar{\mathcal{Q}}_\nu^+$, we can interpret the adjoint operator $-\partial_t + \bar{\mathcal{Q}}_\nu^+$ as another convection-diffusion equation which propagates the adjoint field backwards in time and where the convective flow field (U, V, W) has opposite sign compared to the direct operator $\bar{\mathcal{Q}}_\nu$.

Boundary and initial conditions for this adjoint equation are obtained using the term $\mathcal{J}$, which contains the remainders of the integration by parts:

$$\begin{aligned}\mathcal{J} \;= \int_0^T \int_\Omega & \underbrace{\partial_t\left(uu^+\right)}_{\text{(a)}} + \underbrace{\left(U\partial_x + V\partial_y + W\partial_z\right)\left(u^+u\right)}_{\text{(b)}} \\ & + \underbrace{\nu\left[\partial_x\left(u_x u^+\right) + \partial_y\left(u_y u^+\right) + \partial_z\left(u_z u^+\right)\right]}_{\text{(c)}} - \underbrace{\nu\left[\partial_x\left(u u_x^+\right) + \partial_y\left(u u_y^+\right) + \partial_z\left(u u_z^+\right)\right]}_{\text{(d)}} d\Omega\, dt.\end{aligned} \tag{2.20}$$

Four different terms can be identified in $\mathcal{J}$ with different origins: (a) from the integration of the time derivative, (b) from the integration of the convective term, (c) and (d) from the integration of the diffusive term.

A correct treatment of $\mathcal{J}$ requires the use of the boundary and initial conditions $\mathcal{H}_\mathcal{L}$, $\mathcal{G}_\mathcal{L}$, together with their corresponding inner products. The inner products are defined in (2.16) and have been already used in the definition of the Lagrangian $\mathcal{I}$ (2.15). For example, we consider the term (a): integration in time results in two space integrals evaluated at $t = 0$ and $t = T$. Adding the inner product implementing the initial condition we obtain

$$\int_\Omega uu^+\, d\Omega\Big|_{t=T} - \int_\Omega uu^+\, d\Omega\Big|_{t=0} + \int_\Omega \left(u(t=0) - u_0\right) u_0^+ d\,\Omega.$$

Variation with respect to u is now straightforward and leaves only the first integral evaluated at $t = T$ which has the role — once matched with the variation of the objective functional with respect to u — of defining the initial condition for the adjoint field u^+ at time T. For example, if the objective functional is taken as the energy at time T, i.e. $obj = 0.5\int_\Omega u^H u d\Omega\big|_{t=T}$, the initial condition for the adjoint would be

$$\mathcal{G}_\mathcal{L}^+\left(\mathbf{q}\right) \equiv \qquad u^+\left(T\right) - u\left(T\right) = 0. \tag{2.21}$$

The role of the adjoint field is then related to the enforcement of the optimality condition.

The treatment of the terms $(b), (c), (d)$ of $\mathcal{J}$ (2.20), together with the enforcement of the boundary conditions through the inner product $\left\langle \mathbf{q}_\Gamma^+, \mathcal{H}_\mathcal{L}\mathbf{q}_\Gamma\right\rangle_{\partial\Omega}^T$ — see equation (2.15) — provides the boundary conditions $\mathcal{H}_\mathcal{L}^+$ for the adjoint equation in a similar manner.

All other terms in the $\mathcal{L}$ operator defining the perturbation problem can be dealt with in a similar way: the terms containing the gradients of the base flow remain unchanged while the gradient of the

pressure and the divergence of the perturbations change sign when applied to the adjoint pressure and perturbations, respectively, in a way similar to the change in sign of the time derivative.

The adjoint linearized Navier-Stokes equations for the perturbations then read

$$\mathcal{L}^+ \delta\mathbf{q} = \left(\begin{bmatrix} -\partial_t & & & \\ & -\partial_t & & \\ & & -\partial_t & \\ & & & 0 \end{bmatrix} + \begin{bmatrix} \bar{\mathcal{Q}}_\nu^+ + U_x & U_y & U_z & -\partial_x \\ V_x & \bar{\mathcal{Q}}_\nu^+ + V_y & V_z & -\partial_y \\ W_x & W_y & \bar{\mathcal{Q}}_\nu^+ + W_z & -\partial_z \\ -\partial_x & -\partial_y & -\partial_z & 0 \end{bmatrix} \right) \begin{bmatrix} u^+ \\ v^+ \\ w^+ \\ p^+ \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \quad (\mathcal{L}^+)$$

where $\bar{\mathcal{Q}}_\nu^+$ has been defined in equation (2.19). We describe initial and boundary conditions using the operator $\mathcal{H}_\mathcal{L}^+$ and $\mathcal{G}_\mathcal{L}^+$.

As is the case for the direct equations, all coefficients in the $\mathcal{L}^+$ operator are constant in time and a Laplace transform of the perturbation variables can be performed. A solution of $\mathcal{L}^+$ can be sought as a linear combination of global adjoint modes, each having the form

$$\left(u^+, v^+, w^+, p^+\right) = \left(\hat{u}^+, \hat{v}^+, \hat{w}^+, \hat{p}^+\right) e^{\sigma^+ t} \tag{2.22}$$

where $\sigma^+ = \sigma^H$ is the complex conjugate of σ in (2.12) and $\hat{u}^+, \hat{v}^+, \hat{w}^+, \hat{p}^+$ are complex-valued fields which depend on σ^+ but not on time. As a consequence, the $\mathcal{L}^+$ operator is transformed into $\hat{\mathcal{L}}^+$, and the adjoint linearized Navier-Stokes system can be rewritten as

$$\hat{\mathcal{L}}^+ \hat{\mathbf{q}}^+ = \left(-\sigma \begin{bmatrix} 1 & & & \\ & 1 & & \\ & & 1 & \\ & & & 0 \end{bmatrix} + \begin{bmatrix} \bar{\mathcal{Q}}_\nu^+ + U_x & U_y & U_z & -\partial_x \\ V_x & \bar{\mathcal{Q}}_\nu^+ + V_y & V_z & -\partial_y \\ W_x & W_y & \bar{\mathcal{Q}}_\nu^+ + W_z & -\partial_z \\ -\partial_x & -\partial_y & -\partial_z & 0 \end{bmatrix} \right) \begin{bmatrix} \hat{u}^+ \\ \hat{v}^+ \\ \hat{w}^+ \\ \hat{p}^+ \end{bmatrix} = \begin{bmatrix} u_T^+ \\ v_T^+ \\ w_T^+ \\ 0 \end{bmatrix} \tag{2.23}$$

where the right-hand side contains the initial conditions for the adjoint field.

The homogeneous equivalent of $(\mathcal{L}^+)$, i.e. when the initial conditions $\mathbf{u}_0^+$ are zero, constitutes the adjoint eigenvalue problem

$$\left(-\sigma^+ B^+ + A^+\right) \hat{\mathbf{q}}^+ = 0 \tag{2.24}$$

which can be solved for the adjoint global modes.

Adjoint equations as Hermitian transpose

Now that a physical interpretation of the adjoint field has been given by stating that the initial conditions of the adjoint equations at time $t = T$ are defined by the objective functional, another approach to the computation of the adjoint shall be given. Keeping in mind that, ultimately, we will need to solve our governing equations numerically, the two approaches are shown in Figure 2.2. The first option, represented by continuous arrows, is to first derive the adjoint operator $(\mathcal{L}^+)$ in the manner outlined in the previous section, and then proceed to its discretization. This approach requires to compute the adjoint system and to perform two different discretizations — of the direct and the adjoint system. The second option, represented by dashed arrows, starts by implementing a discretization $\mathcal{L}_h$ of the linear operator $\mathcal{L}$, including all boundary conditions. The discrete adjoint operator $\mathcal{L}_h^+$ can then be obtained as the Hermitian transpose of the direct, thus bypassing the computation of the continuous adjoint equations

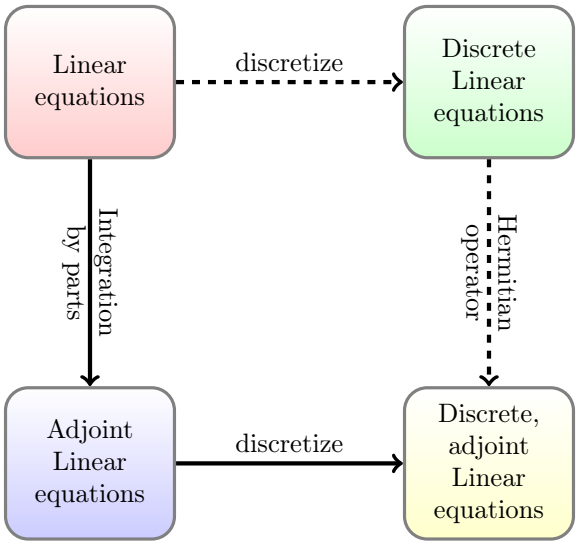


Figure 2.2: Possible routes to obtain the adjoint equations. Continuous arrows: starting from the direct linear problem (red), the adjoint equations can be computed by integration by parts (blue) and can then be discretized (yellow). Alternatively, the dashed arrows show that if discretization is performed on the linear problem first (green), the adjoint equations can be obtained by simply computing the Hermitian transpose of the discrete problem.

and their discretization.

$$\mathcal{L}_h^+ = \mathcal{L}_h^H$$

The same relationship hold for the Laplace transformed operator

$$\hat{\mathcal{L}}_h^+ = \hat{\mathcal{L}}_h^H.$$

As a consequence, we can reinterpret the adjoint eigenvectors as the left eigenvectors of the direct problem.

2.4 Receptivity and sensitivity

The linearized Navier-Stokes operator ($\mathcal{L}$) is a non-normal operator, where the non-normality is mainly the consequence of the streamwise advection term $\left(U\partial_x + V\partial_y + W\partial_z\right)$ in the $\bar{\mathcal{Q}}_\nu$ operator acting in the opposite direction on the adjoint field. For a comprehensive review of operator non-normality the reader is referred to Schmid & Henningson [61].

Operator non-normality has three principal consequences [12]. First, the perturbation energy $\int_\Omega \mathbf{u}\cdot\mathbf{u}\, d\Omega$ can exhibit transient growth, i.e. an increase — possibly of several orders of magnitude — in energy over a short time even if the system is asymptotically stable. This mechanism can be at the origin of by-pass transition, where transition from laminar to turbulent fluid motion is not related to the asymptotic exponential growth of the perturbations but rather to the transient growth over a short time interval. The amount of transient growth depends on the initial condition, and the search for the specific initial condition maximizing growth over a certain time interval leads to the optimal perturbation problem. The solution of the optimal perturbation problem is based on the maximization of a Lagrangian functional similar to (2.15) where the objective functional obj is, in the simplest case, given by the energy amplification of the perturbations at a given time or the integral of the energy within a given time interval. The

optimal perturbation problem has been extensively studied for the case of swept Hiemenz flow, a flow configuration closely related to ours, by Guégan, Schmid & Huerre [23, 22, 24] and will not be addressed here. The potential of swept Hiemenz flow to support transient growth has been analyzed by Obrist & Schmid [49].

The other two consequences of non-normality are related to the long time (or asymptotic) response: a very strong receptivity/response to forcing and a marked sensitivity of the spectrum — which can be obtained by solving the eigenvalue problem (2.14) — to perturbations of the operator ($\mathcal{L}$). These two problems will be addressed in the following sections employing the Lagrangian framework already used in the derivation of the adjoint.

Receptivity to forcing

We start by slightly modifying the definition of the Lagrangian functional (2.15) by including the forcing of the perturbation equations, which so far has been considered as user-specified, as a variable. Additionally, the Laplace-transformed operator $\hat{\mathcal{L}}$ and state variables $\hat{\mathbf{q}}$ are used as constraints instead of the original operator $\mathcal{L}$ and variables $\mathbf{q}$. This is equivalent to taking the Laplace transform of $\mathcal{I}$, provided that the initial conditions are multiplied by a Dirac delta-function δ. We have

$$\hat{\mathcal{I}}\left(\hat{\mathbf{q}},\hat{\mathbf{q}}^+,\hat{\mathbf{q}}_0,\hat{\mathbf{q}}_0^+,\hat{\mathbf{q}}_\Gamma,\hat{\mathbf{q}}_\Gamma^+,\hat{\mathbf{f}}'\right) = obj - \left\langle\hat{\mathbf{q}}^+,\hat{\mathcal{L}}\left(\mathbf{Q}\right)\hat{\mathbf{q}}-\hat{\mathbf{f}}'\right\rangle_\Omega - \left\langle\hat{\mathbf{q}}_0^+,\mathcal{G}_\mathcal{L}\hat{\mathbf{q}}_0\right\rangle_\Omega - \left\langle\hat{\mathbf{q}}_\Gamma^+,\hat{\mathcal{H}}_\mathcal{L}\hat{\mathbf{q}}_\Gamma\right\rangle_{\partial\Omega}. \tag{2.25}$$

Taking the first variation with respect to the adjoint and direct fields gives the direct (2.13) and adjoint (2.23) governing equations respectively, together with boundary conditions. In addition, we must now consider the variation of the Lagrangian with respect to the forcing itself, again using the definition of the functional derivative given in (2.17). The only terms contributing to the variation are the objective functional — which we assume depends on the forcing — and the first inner product. The change in the objective functional associated with a change in the forcing $\delta\left(obj\right) = \partial\left(obj\right)/\partial\hat{\mathbf{f}}'\,\delta\hat{\mathbf{f}}'$ is obtained by setting this variation to zero. We obtain

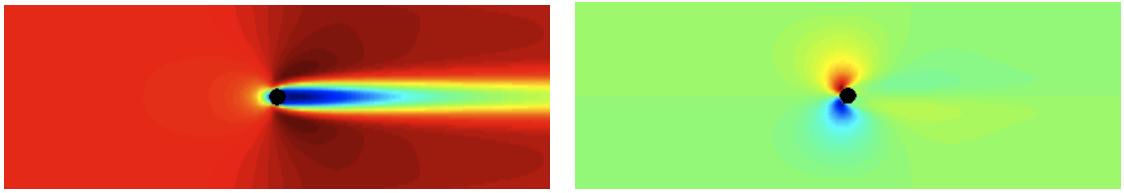
$$\frac{\partial\hat{\mathcal{I}}}{\partial\hat{\mathbf{f}}'}\delta\hat{\mathbf{f}}' = \frac{\partial\left(obj\right)}{\partial\hat{\mathbf{f}}'}\delta\hat{\mathbf{f}}' + \left\langle\hat{\mathbf{q}}^+,\delta\hat{\mathbf{f}}'\right\rangle_\Omega = 0 \qquad \Longrightarrow \qquad \delta\left(obj\right) = -\left\langle\hat{\mathbf{q}}^+,\delta\hat{\mathbf{f}}'\right\rangle_\Omega \tag{2.26}$$

which states that the sensitivity to small changes in the external forcing $\hat{\mathbf{f}}'$ is simply given by the negative of the adjoint field $\hat{\mathbf{q}}^+$.

In order to exemplify the consequences of these results, we consider the case of the two-dimensional wake developing downstream of a cylinder placed in a uniform flow, previously studied by Giannetti & Luchini [18, 19].

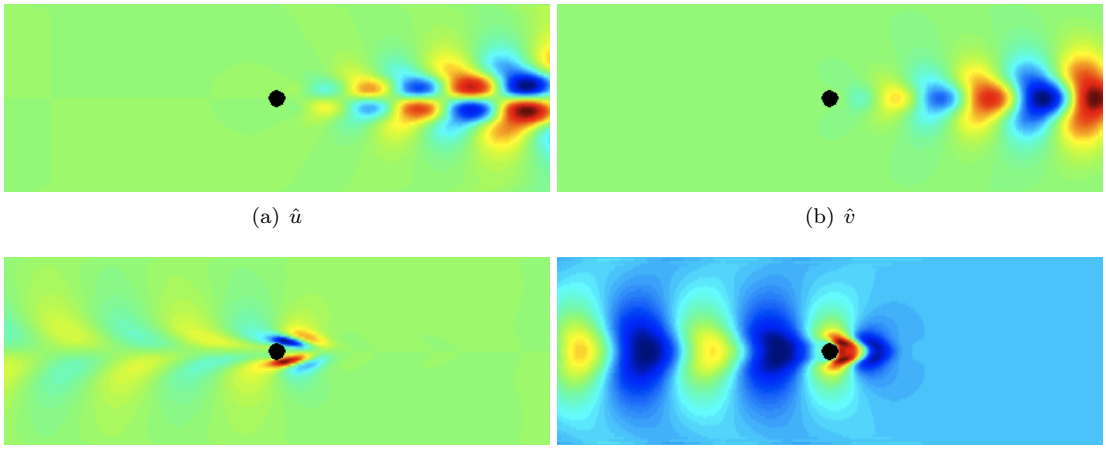
The base flow for this configuration is shown in Figure 2.3, visualized by its U and V velocity components. The flow is from left to right and is characterized by a recirculation bubble extending a few cylinder diameters downstream of the cylinder, as shown by the blue area in the U velocity field. The $\hat{u}$ (a and c) and $\hat{v}$ (b and d) velocity components of the direct (a and b) and adjoint (c and d) most unstable global modes are represented in Figure 2.4. The spatial separation of the direct and adjoint modes, which is itself a consequence of the operator non-normality, is evident: the direct mode is concentrated in the downstream part of the domain (for Reynolds numbers close to the critical, diameter-based Reynolds number of ≈ 47, the direct mode represents the von Karman vortex street) while the adjoint mode is mainly concentrated upstream and in the separation bubble developing just downstream of the cylinder.

Following the interpretation of the adjoint field as the gradient of the objective functional with respect to the forcing, Figure 2.4 provides useful information on where in the domain it is most effective to apply



(a) U (b) V

Figure 2.3: U, V velocity components for flow around a cylinder, courtesy of P.J. Schmid.



(a) $\hat{u}$ (b) $\hat{v}$

(c) $\hat{u}^+$ — receptivity to stream-wise forcing (d) $\hat{v}^+$ — receptivity to cross-stream forcing

Figure 2.4: Direct and adjoint fields for a cylinder wake, courtesy of P.J. Schmid.

a forcing in order to modify the objective functional: a localized forcing in the stream-wise direction will be most effective in changing the objective functional where the u^+ adjoint eigenvector reaches its maximum, while forcing in the y direction will be most effective where the v^+ adjoint eigenvector reaches its maximum. In both cases, the location is slightly downstream of the cylinder, close to the point where the flow detaches from the cylinder's surface forming the recirculation bubble.

The objective functional itself remains to be specified, and it can be shown [19] that in this case it corresponds to the amplitude of the eigenvector.

Sensitivity of the spectrum to structural modifications

We now want to address the issue of sensitivity of the spectrum to a structural change in the operator. This problem has been first addressed by Giannetti and Luchini [18, 19].

In order to understand how a given eigenvalue σ changes as the governing operator $\hat{\mathcal{L}}$ changes, we first rewrite $\hat{\mathcal{L}}$ by separating the complex frequency σ from the spatial part, as already seen when we defined the generalized eigenvalue problem (2.14)

$$\hat{\mathcal{L}} = (\sigma B + A)\,,$$

and consider a variation δA in the spatial part A of the operator. Examples of the possible origin of such a variation δA are a change in the Reynolds number, a change in the base flow field or, if the boundary

conditions are included in $\mathcal{L}$, as is the case after the problem is discretized, a change in the boundary conditions. Once again, we start with the definition of the Lagrangian $\mathcal{I}$, with the difference that we now include the operator A as a variable. Additionally, the objective functional is specified as the eigenvalue σ, itself a function of A and the sensitivity of which we are interested in; the forcing in the perturbation equations is set to zero. The new Lagrangian reads

$$\hat{\mathcal{I}}\left(\sigma, \hat{\mathbf{q}}, \hat{\mathbf{q}}^+, \ldots, A\right) = \sigma - \left\langle \hat{\mathbf{q}}^+, (\sigma B + A)\,\hat{\mathbf{q}} - \hat{\mathbf{f}}' \right\rangle_\Omega - \ldots \tag{2.27}$$

where we have omitted the inner products related to the initial and boundary conditions as their manipulation is similar to what has been done before. Alternatively, we can assume the direct and adjoint fields $\hat{\mathbf{q}}$ and $\hat{\mathbf{q}}^+$ to be defined also on the boundaries and the A and B operator to take this fact into account. This latter approach is natural when dealing with the discretized operator.

As already seen in the derivation of the adjoint perturbation equations, variations of the Lagrangian functional with respect to its variables are obtained by applying the definition of the functional derivative (2.17). The first variation with respect to the adjoint state variables $\partial\mathcal{I}/\partial\hat{\mathbf{q}}^+$ returns the direct operator $\hat{\mathcal{L}} = (\sigma B + A)$ while the first variation with respect to the direct state variables $\partial\mathcal{I}/\partial\hat{\mathbf{q}}$ returns the adjoint operator $\hat{\mathcal{L}}^+ = \left(-\sigma B + A^H\right)$.

In addition, we have to compute variations with respect to the two new variables σ and δA. Computing the variation with respect to σ is straightforward and setting it to zero provides the normalization condition for the adjoint eigenvector according to

$$\frac{\partial\hat{\mathcal{I}}}{\partial\sigma}\delta\sigma = \delta\sigma - \left\langle \hat{\mathbf{q}}^+, B\hat{\mathbf{q}} \right\rangle_\Omega \delta\sigma = 0 \qquad \Longrightarrow \qquad \left\langle \hat{\mathbf{q}}^+, B\hat{\mathbf{q}} \right\rangle_\Omega = 1. \tag{2.28}$$

It now remains to compute the variation with respect to the structural modification δA. In performing this operation we have to take into account the fact that the eigenvalue σ is itself a function of the operator A and has to be treated accordingly. The change of the eigenvalue associated with the change in the operator $\delta\sigma = \partial\sigma/\partial A\;\delta A$ is obtained by setting this variation to zero.

$$\frac{\partial\hat{\mathcal{I}}}{\partial A}\delta A = \frac{\partial\sigma}{\partial A}\delta A - \left\langle \hat{\mathbf{q}}^+, \delta A\,\hat{\mathbf{q}} \right\rangle = 0 \qquad \Longrightarrow \qquad \delta\sigma = \left\langle \hat{\mathbf{q}}^+, \delta A\,\hat{\mathbf{q}} \right\rangle_\Omega \tag{2.29}$$

The role of non-normality in increasing the sensitivity of the spectrum to structural modifications δA can be further illustrated by considering the normalization condition (2.28): the spatial separation between the direct and adjoint modes seen in Figure 2.4 requires large values of the adjoint eigenvector for the integral over the area where the two coexist to be one.

The wavemaker

Once we have obtained the sensitivity of the spectrum to structural modifications δA of the operator, we can consider a localized structural modification and determine where in the domain it would be most effective.

The answer to this question has implications for the design of a control strategy and often represents the first step of a control study by identifying the location where a localized feedback of the perturbations onto themselves is most effective. An example of such a control strategy has been considered by Giannetti & Luchini [19] for the case of a cylinder wake: they considered a feedback process in the form of a forcing $\mathbf{f}' = \mathbf{C}(\mathbf{x})\,\mathbf{u}$ where the matrix $\mathbf{C}(\mathbf{x})$ identifies the linear dependence of the forcing $\mathbf{f}'$ on the perturbation velocity field $\mathbf{u}$. They additionally consider the feedback process to be localized at a given position $\mathbf{x_0}$ in

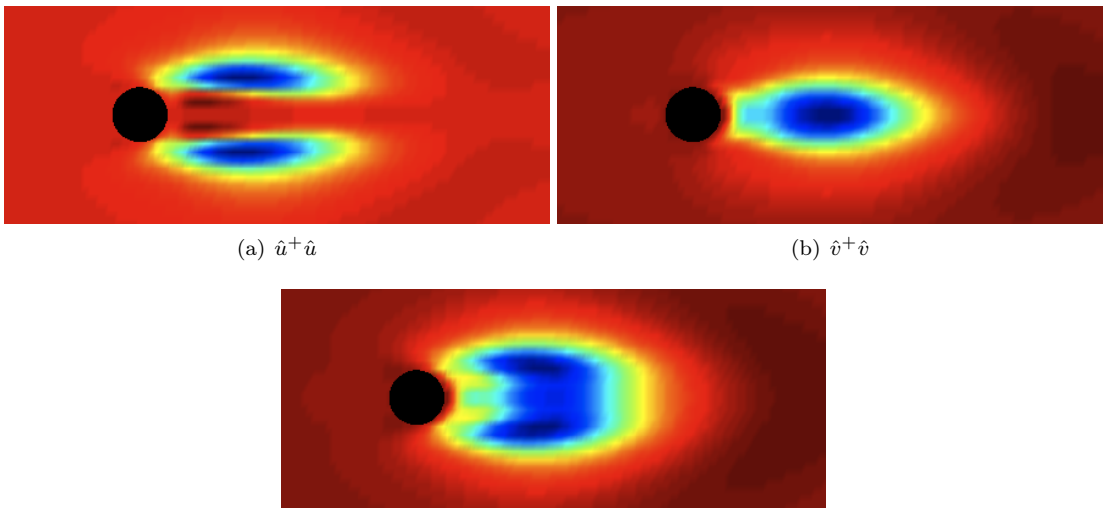
the domain by envisaging the special case $\mathbf{C}(\mathbf{x}) = \mathbf{C_0}\delta(\mathbf{x} - \mathbf{x_0})$, where $\delta(\mathbf{x} - \mathbf{x_0})$ is the Kronecker delta function.

The time independence of the matrix $\mathbf{C}(\mathbf{x})$ suggests a modal representation of both the forcing and the perturbation field, similar to what has been done before: the application of the corresponding Laplace transform allows us to consider $\mathbf{C_0}$ as the perturbation δA of the operator in (2.29), under the hypothesis that $\mathbf{C_0}$ is small compared to A. As a consequence, we can rewrite the shift of the eigenvalue starting from (2.29) as

$$\delta\sigma = \left\langle \hat{\mathbf{q}}^+, \mathbf{C_0}\delta(\mathbf{x} - \mathbf{x_0})\ \hat{\mathbf{q}} \right\rangle_\Omega. \tag{2.30}$$

From an application point of view, this formulation can describe the placement of a small cylinder some place in the domain exerting a forcing $\mathbf{f}' = -c_d\mathbf{u}$ on the perturbations, where c_d is a drag coefficient. It should be mentioned that this description limits us to a forcing described by Stokes flow, i.e. with a linear dependence on the velocity and without any intrinsic dynamics, as it immediately responds to any modification of the perturbation field $\mathbf{u}$.

We can now answer the question asked at the beginning of this section: what is the position where a small cylinder — that is, our localized structural modification — would be most effective? The answer is provided by analyzing the shift of the eigenvalue $\delta\sigma$ as a function of the location of the forcing $\mathbf{x_0}$ as given by (2.30) and plotted for the case of the cylinder wake in Figure 2.5, where the point-wise product of the adjoint and the direct field for the $\hat{u}$ and $\hat{v}$ velocity components, as well as for the magnitude of the velocity field, are displayed.



(a) $\hat{u}^+\hat{u}$ (b) $\hat{v}^+\hat{v}$

(c) $|\hat{\mathbf{u}}^+\hat{\mathbf{u}}|$

Figure 2.5: Contours of the wavemaker regions, courtesy of P.J. Schmid, based on the work of Giannetti & Luchini [19]

A dual point of view interprets the same result as identifying the wavemaker of the global mode, defined as the region where *a modification in the structure of the problem is able to produce the greatest drift of the eigenvalue* [19]. From equation (2.30) it can be seen that changes in the operator have a significant effect on the eigenvalues in regions where the direct and adjoint global modes substantially overlap. Outside the region of marked overlap, a rather small effect on the position of the eigenvalue is observed. As has been shown in the case of the cylinder wake [19] and as we will show for the case of attachment-line flow, a consequence of this observation is that, in an eigenvalue computation, only the

wavemaker region needs to be numerically represented in order to obtain the correct spectrum and the representation of the operator outside this region is of less relevance.

As a last remark, we note that there is no need for a small cylinder in order to observe a feedback process. A perturbation δA in the operator can be identified as a change in the base flow, itself associated with the presence of the perturbation, as the perturbed flow is given by the sum $\mathbf{Q} + \mathbf{q}$. In this sense, the wavemaker identifies the region where the coupled effect of perturbations and receptivity is strongest: large perturbations where the receptivity is low do not affect the flow behavior, strong receptivity in regions where perturbations are insignificant does not affect it either.

Chapter 3

Numerical Approach

A thorough investigation of the problem described in chapter 2 requires the computation of discrete solutions of the nonlinear Navier-Stokes equations ($\mathcal{R}$) and of the direct and adjoint eigenvalue problems associated with the linearized Navier-Stokes equations ($\mathcal{L}$). The present and the next chapter are devoted to the numerical approach used in this work. In this chapter the discrete representation of the nonlinear and linear governing equations is introduced and validated. In the next chapter we describe the multigrid algorithm used in the solution of the discretized, nonlinear Navier-Stokes equations ($\mathcal{R}$).

The formulation of the Navier-Stokes equations ($\mathcal{R}$) given in chapter 2, which we refer to as the divergence form of the Navier-Stokes equations, is not the most suitable from a computational point of view. An alternative formulation, referred to as the pressure form, is presented in the first section of this chapter [21, 20, 62, 56]. It is used in all computations performed in this work and can be obtained by applying a projection operator $\mathcal{P}$ to the divergence form ($\mathcal{R}$). The application of $\mathcal{P}$ can be related to the computation of the divergence of the momentum equations. Provided that the correct boundary conditions for the pressure field are used, the solutions based on the two formulations are identical: in the remainder of this work we then simply refer to the solution of the Navier-Stokes equations, independently of the formulation used to compute it.

We find it useful to specify from the start that the pressure form of the Navier-Stokes system must not be confused with the projection/fractional-step approach often used in time-stepping algorithms, despite some similarities like the introduction of a Laplacian operator applied to the pressure field.

In order to compute a discrete representation of our flow field, the pressure form of the Navier-Stokes equations is discretized on a grid obtained by conformally mapping a rectangle of size l_ξ, l_η to the domain surrounding a Joukowsky profile. The domain used for the computation of the base flow covers approximately 20% of the chord, while smaller domains are used for the computation of the eigenvalues and eigenvectors. A second-order finite difference discretization of the governing equations is defined on the conformally mapped grid. Upwinded stencils are used for the convective terms and centered stencils for all other first-order derivatives The Laplacian operator is discretized using a finite-volume like formulation. Additionally, the spanwise invariance of the base flow allows for some simplifications in the governing equations and the application of a Fourier transform in the spanwise z-direction when dealing with the governing equations for the perturbations ($\mathcal{L}$).

The implemented discretization of both the nonlinear and the linear problem needs to be validated, and a set of tests are performed in order to verify its correctness. Two tests are performed for the nonlinear problem: (i) the computation of the discrete residual of the analytical solution for the inviscid flow around a Joukowsky profile is used to verify that the discretization in the interior of the domain is second order with respect to the mesh size h and (ii) the solution of the viscous flow field around a cylinder is used to verify the implementation and discretization of the boundary conditions by comparing our results with results from the literature and from an alternative, well-established numerical code. Validation of the linearized equations is performed by comparing the implemented discretization of the operator $\mathcal{L}$ with a

finite-difference approximation computed from the nonlinear equations $\mathcal{R}$.

At the end of this chapter we discuss the issue related to the presence of different scales in the problem. As a result, the discretized problem becomes very large when increasing the Reynolds number if a global approach is pursued. Useful methods in overcoming this difficulty, including grid stretching and adaptive refinement, will be introduced in the next chapter, but some considerations on the expected size of the system will be presented as well.

3.1 The pressure form of the Navier-Stokes equations

The formulation of the Navier-Stokes equations ($\mathcal{R}$) given in chapter 2, which we refer to as the divergence form, is not the most appropriate formulation that can be used for the computation of a numerical solution. Centered differencing of the continuity equation $\nabla \cdot \mathbf{u}$ and the pressure gradient ∇p on a collocated grid can result in spurious oscillations of the solution and can be a source of slow convergence rates in the multigrid solver that will be described in chapter 4 [64].

Common approaches used to overcome these difficulties include staggered grid discretization, where the momentum equations and the continuity equation are defined in different locations in a computational cell, and projection/fractional step methods, where advancement in time of the momentum equations and the enforcement of the continuity equation are treated in two separate steps. While both these approaches are valid, they also come with their drawbacks: staggered grids are rarely used for complex geometries as they require the handling of a rather complicated grid structure; fractional step methods require the use of a time-stepping or pseudo-time-stepping algorithm whose convergence to a steady state solution can be rather slow and, in the simplest case, requires multiple solutions of a Poisson equation — for the velocity and the pressure — at each time step.

An alternative and more effective approach, in particular in the context of the multigrid framework, is the use of a pressure formulation — or pressure Poisson formulation — of the Navier-Stokes equations [21, 20, 62, 56] The pressure formulation can be obtained by left-multiplying both sides of the divergence form of the Navier-Stokes system ($\mathcal{R}$) by the operator

$$\mathcal{P} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ \partial_x & \partial_y & \partial_z & -\mathcal{Q}_\nu \end{bmatrix} \tag{3.1}$$

where $\mathcal{Q}_\nu = u\partial_x + v\partial_y + w\partial_z - \nu\left(\partial_{xx} + \partial_{yy} + \partial_{zz}\right)$ is a nonlinear operator representing the advection and diffusion terms [65]. This operation leaves the momentum equations unchanged and replaces the continuity equation with a Poisson equation for the pressure in the form

$$\nabla \cdot \left(\mathcal{Q}_\nu \mathbf{u} + \nabla p\right) - \mathcal{Q}_\nu \left(\nabla \cdot \mathbf{u}\right) = \nabla \cdot \mathbf{f} \tag{3.2}$$

where for simplicity we have used the steady-state version of the Navier-Stokes operator $\mathcal{R}$. The first term of the left-hand side represents the divergence of the momentum equations and the second term is a convection-diffusion equation for the divergence field. The right-hand side is given as the divergence of the forcing.

The resulting set of equations constitutes the pressure formulation of the Navier-Stokes system and

has a Poisson equation for the pressure in lieu of the continuity equation:

$$\mathcal{R}_\Delta(\mathbf{q}) \equiv \begin{cases} \partial_t \mathbf{u} + \nabla \mathbf{u}\,\mathbf{u} - \nu\Delta\mathbf{u} + \nabla p &= \mathbf{f} \\ \mathcal{Q}_p + \Delta p &= m_\Delta \end{cases} \tag{$\mathcal{R}_\Delta$}$$

where $\mathcal{Q}_p = (u_x)^2 + (v_y)^2 + (w_z)^2 + 2v_x u_y + 2u_z w_x + 2v_z w_y$ stands for the nonlinear term containing gradients of the velocity field. The term $m_\Delta = \nabla \cdot \mathbf{f}$ is the divergence of the forcing $\mathbf{f}$.

The application of the operator $\mathcal{P}$ to the linearized Navier-Stokes system ($\mathcal{L}$) — or, alternatively, the linearization of the pressure Poisson formulation ($\mathcal{R}_\Delta$) — produces a linearized pressure Poisson equation; the corresponding linearized Navier-Stokes equations read

$$\mathcal{L}_\Delta \delta\mathbf{q} \equiv \left(\begin{bmatrix} \partial_t & & & \\ & \partial_t & & \\ & & \partial_t & \\ & & & 0 \end{bmatrix} + \begin{bmatrix} \bar{\mathcal{Q}}_\nu + U_x & U_y & U_z & \partial_x \\ V_x & \bar{\mathcal{Q}}_\nu + V_y & V_z & \partial_y \\ W_x & U_y & \bar{\mathcal{Q}}_\nu + W_z & \partial_z \\ \mathcal{Q}_1 & \mathcal{Q}_2 & \mathcal{Q}_3 & \Delta \end{bmatrix} \right) \begin{bmatrix} u \\ v \\ w \\ p \end{bmatrix} = \begin{bmatrix} f'_u \\ f'_v \\ f'_w \\ m'_\Delta \end{bmatrix} \tag{$\mathcal{L}_\Delta$}$$

where $\bar{\mathcal{Q}}_\nu$ denotes the same expression as defined in equation (2.9):

$$\bar{\mathcal{Q}}_\nu = U\partial_x + V\partial_y + W\partial_z - \nu\left(\partial_{xx} + \partial_{yy} + \partial_{zz}\right), \tag{3.3a}$$

$$\mathcal{Q}_1 = -2\left(V_y + W_z\right)\partial_x + 2V_x\partial_y + 2W_x\partial_z, \tag{3.3b}$$

$$\mathcal{Q}_2 = +2U_y\partial_x - 2\left(U_x + W_z\right)\partial_y + 2W_y\partial_z, \tag{3.3c}$$

$$\mathcal{Q}_3 = +2U_z\partial_x + 2V_z\partial_y - 2\left(U_x + V_y\right)\partial_z. \tag{3.3d}$$

As noted in the previous chapter, adjoint equations can be obtained by computing the complex conjugate (Hermitian) transpose of the linearized system ($\mathcal{L}_\Delta$). Computation of the Laplace transform and of the eigenvalue problems associated with the direct and adjoint operators in pressure form does not differ substantially from the case of the operators in divergence form and is hence omitted here.

Boundary conditions for the pressure form

The use of the pressure equation in place of the continuity equation poses two problems. (i) Whether and under which conditions are the two introduced formulations equivalent, i.e., will the pressure equation correctly enforce continuity? And (ii) what is the additional boundary condition for the pressure, which is now required along the entire boundary of the computational domain due to the introduction of a Δp term in the equations — see Sani et al. [56] for a case in which it is not required.

The problem posed by these two issues is still open, and we refer to the work of Gresho, Sani and co-authors [21, 20, 56] for a discussion on the subject. Nonetheless, Swanson [64] successfully applied the continuity equation as a boundary condition for the pressure in order to enforce a divergence-free flow field in a geometry similar to ours. The two questions above seem to be related, and imposing the correct boundary conditions results in the continuity equation being satisfied everywhere in the domain. This is the approach used in this work.

We find it important to remark at this point that a homogeneous Neumann boundary condition $\partial p/\partial n = 0$ is often applied to the pressure field in the context of fractional step methods. Our numerical experiments have shown that this boundary condition fails to enforce continuity when a solution is sought for the steady-state pressure form of the Navier-Stokes equations, resulting in unrealistically thick

boundary layers where mass is generated or destroyed.

The implementation of the continuity equation as a boundary condition for the pressure relies on reformulating the momentum equations projected in the direction normal to the boundary, such that a finite-volume like formulation of the viscous terms can be used. To exemplify this procedure we consider

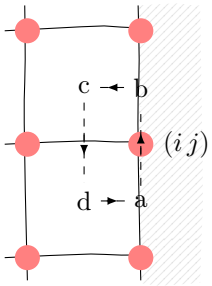


Figure 3.1: Solid boundary control cell

a two-dimensional problem with a solid boundary aligned with the vertical y-axis as shown in Figure 3.1, so that the normal momentum equation is the u-momentum equation. We can then express the normal pressure gradient as

$$\frac{\partial p}{\partial n} = \frac{\partial p}{\partial x} = -\left(uu_x + vu_y\right) + \nu\Delta u. \tag{3.4}$$

Because of the no-slip condition, the convective part $uu_x + vu_y$ is identically zero. After rewriting the Laplacian operator using the divergence theorem as $\Delta u = 1/\Omega \int_{\partial\Omega} \nabla u \cdot \mathbf{n} ds$, with the control cell Ω outlined by dashed lines in Figure 3.1, we can write

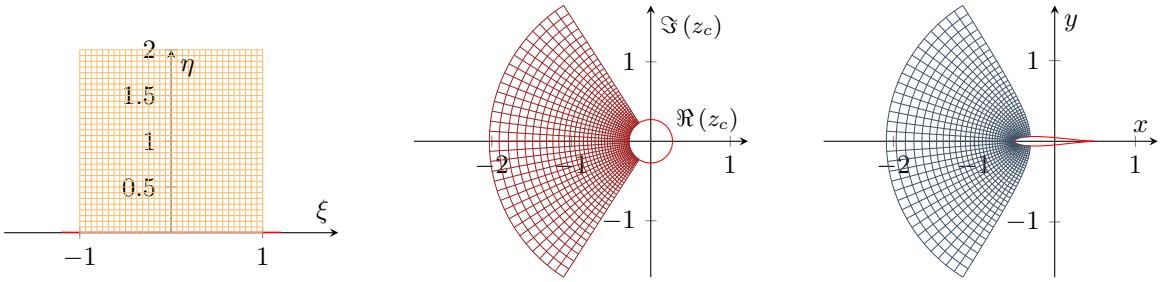
$$\frac{\partial p}{\partial n} = \nu \int_{\partial\Omega} \nabla u \cdot \mathbf{n} ds = \nu \int_{\partial\Omega} u_x dy - u_y dx \tag{3.5}$$

where the integration is performed on $abcd$. On the solid boundary ab the continuity equation is applied and u_x is replaced by $-v_y$, which is zero because of the no-slip condition. The normal pressure gradient so obtained is used as a boundary condition for the pressure [64].

3.2 Computational domain and discretization

In order to compute a discrete representation of our flow field, the pressure form of the Navier-Stokes equations is discretized on a grid obtained by conformally mapping a rectangle of size l_ξ, l_η onto a domain surrounding a Joukowsky profile. The domain used for the computation of the base flow covers roughly 20% of the chord and is shown on the right of Figure 3.2; smaller domains, whose extension will be specified later, are used for the subsequent eigenvalue computations. Taking advantage of the spanwise invariance hypothesis, a Fourier transform is applied in the spanwise z-direction: all first-order z-derivatives are replaced by ik_z and all second-order z-derivatives by $-k_z^2$, where k_z denotes the spanwise wave number. For the base flow computation, we take $k_z = 0$.

The conformal mapping is performed in two steps as shown in Figure 3.2: (i) the rectangular domain is mapped onto a circular sector using the complex exponential function and (ii) the circular sector is then mapped onto the leading-edge region of a Joukowsky profile. The conformal mapping is used because it represents a straightforward and efficient way of obtaining an orthogonal, body-fitted grid. The parameters that define the domain size and the airfoil shape are: the l_ξ and l_η extension of the



(a) $z_r = \xi + i\,\eta$ (b) $z_c = r_{min}\, e^{z_r}$ (c) $z_J = (z_c - \mu) + \dfrac{C^2}{16\,(z_c - \mu)}$

Figure 3.2: Conformal mapping from a rectangle to the Joukowsky profile: a rectangular domain of size l_ξ, l_η (a) is conformally mapped to a sector of a circle (b) using a complex exponential function. As a consequence, the ξ coordinate becomes the angular coordinate and the η coordinate becomes the radial coordinate. An additional conformal mapping is applied to transform the circle sector into the leading edge of the profile (c). The exponential function alone is at the origin of the exponential grid stretching in the normal (radial) coordinate in figure (b) and (c). An additional stretching defined in equation (3.6) is applied before mapping the rectangle to the circle. The parameters used in the mappings are listed in Table 3.1.

numerical domain, the chord C of the Joukowsky profile and a parameter ϵ governing the profile thickness. The values of r_{min} and μ appearing in the mapping functions are computed as shown in Table 3.1. In addition to the conformal mapping, a stretching function is applied in the η-direction before the first transformation from the rectangular domain to the circular sector. This stretching function reads

$$\bar{\eta} = \frac{k\eta}{1 - k + \eta\,(2k - 1)}. \tag{3.6}$$

Depending on the value of k, the mesh spacing in the domain $[0:1]$ is deformed such that half the points are clustered within the interval $[0:k]$. Thus, $k = 0.5$ corresponds to no stretching and $k < 0.5$ clusters points towards the solid boundary. A value of $k = 0.1$ is used for the computations in this thesis.

Table 3.1: Parameters governing the dimension and shape of the numerical domain used in the computation of the base flow. The corresponding domain is visualized in Figure 3.2.

Parameter	Value	Meaning
l_ξ	2	length of the rectangular domain in the ξ direction
l_η	2	length of the rectangular domain in the η direction
C	1	chord of the Joukowsky profile
ϵ	0.1	parameter governing the thickness of the Joukowsky profile
r_{min}	$-\frac{(1+\epsilon)C}{4}$	radius of the circle in Figure 3.2 (b)
μ	$-\frac{\epsilon\,C}{4}$	parameter governing the shape of the Joukowsky profile[a]
k	0.1	parameter governing the stretching function (3.6)

[a] if μ has a nonzero complex part, camber is added to the Joukowsky profile

The three transformations — stretching, conformal mapping to a circle, and conformal mapping to

the Joukowsky profile — are applied in a cascade and can be summarized as a single transformation represented by the functions $x = x(\xi, \eta)$ and $y = y(\xi, \eta)$, mapping directly the equally spaced Cartesian grid onto the Joukowsky domain. Once the ξ, η and x, y coordinates are known the corresponding transformation matrix can be computed numerically and reads

$$\begin{bmatrix} \xi_x & \eta_x \\ \xi_y & \eta_y \end{bmatrix} = \frac{1}{J} \begin{bmatrix} y_\eta & -y_\xi \\ -x_\eta & x_\xi \end{bmatrix} \tag{3.7}$$

where the metric tensor's Jacobian is $J = x_\xi y_\eta - x_\eta y_\xi$. The mapping derivatives are discretized using centered second-order finite-difference stencils.

In the discretization process, all operators appearing in the Navier-Stokes equations ($\mathcal{R}_\Delta$) are expressed on the Cartesian, equispaced grid in Figure 3.2 (a) and the mapping is taken into account using metric coefficients. Taking into account the Fourier transform in the spanwise z-direction, the convective terms can be rewritten as

$$\left(u\partial_x + v\partial_y + w\partial_z\right) u = \left(\tilde{u}\partial_\xi + \tilde{v}\partial_\eta + \tilde{w}ik_z\right) u \tag{3.8}$$

where $\tilde{u} = \xi_x u + \xi_y v$, $\tilde{v} = \eta_x u + \eta_y v$ and $\tilde{w} = w$ are the contravariant velocity components [64]. The derivatives ∂_ξ and ∂_η are discretized on the Cartesian, equispaced grid using a second-order upwinded discretization, where upwinding is performed with respect to the contravariant velocity components. The pressure gradient in the momentum equations and the velocity derivatives in the pressure equation are rewritten in body-fitted coordinates as

$$\partial_x = \xi_x \partial_\xi + \eta_x \partial_\eta \tag{3.9a}$$

$$\partial_y = \xi_y \partial_\xi + \eta_y \partial_\eta \tag{3.9b}$$

and $\partial_z = ik_z$. ∂_ξ and ∂_η are discretized using centered second-order finite-difference stencils.

As previously mentioned, the Laplacian operator is reformulated using a finite-volume-like form

$$\Delta\phi = \left(\partial_{xx} + \partial_{yy} + \partial_{zz}\right)\phi = \frac{1}{A}\int_{\partial A} \phi_x dy - \phi_y dx \; - k_z^2 \phi \tag{3.10}$$

where the integral is performed over the boundary ∂A of the control cell represented in Figure 3.3. On a Cartesian grid, this formulation corresponds exactly to a finite-difference discretization. The vector $(dy, -dx)$ represents the normal to the boundary ∂A pointing outward of the cell. Both the derivatives and the normal vector can be rewritten for the Cartesian grid and the Laplacian reads

$$\begin{aligned} &\frac{1}{A}\int_{\partial A} \begin{bmatrix} \phi_\xi & \phi_\eta \end{bmatrix} \underbrace{\begin{bmatrix} \xi_x & \xi_y \\ \eta_x & \eta_y \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} y_\eta & y_\xi \\ x_\eta & x_\xi \end{bmatrix}} \begin{bmatrix} d\eta \\ d\xi \end{bmatrix} - k_z^2\phi = \\ =&\frac{1}{A}\int_{\partial A} \begin{bmatrix} \phi_\xi & \phi_\eta \end{bmatrix} \begin{bmatrix} a_{11} & a_{21} \\ a_{12} & a_{22} \end{bmatrix} \begin{bmatrix} d\eta \\ d\xi \end{bmatrix} - k_z^2\phi = \\ =&\frac{1}{A}\int_{\partial A} \left(\phi_\xi a_{21} d\xi + \phi_\eta a_{22} d\xi + \phi_\xi a_{11} d\eta + \phi_\eta a_{12} d\eta\right) - k_z^2\phi \end{aligned}$$

The coefficients a_{ij} are functions of the metric coefficients only and have to be computed on the boundary of the control cell, not on the mesh points.

When discretization is performed, the derivatives and the normals are considered constant on each side of the rectangle *abcd* in Figure 3.3; the integral can be replaced with a summation on the four sides of the rectangle *abcd*, and the expression for the Laplacian simplifies to

$$\Delta\phi = \frac{1}{Jd\xi d\eta} \quad \Big(\left(\phi_\xi a_{21} d\xi + \phi_\eta a_{22} d\xi\right)_{ab} + \left(\phi_\xi a_{11} d\eta + \phi_\eta a_{12} d\eta\right)_{bc} \\ - \left(\phi_\xi a_{21} d\xi + \phi_\eta a_{22} d\xi\right)_{cd} - \left(\phi_\xi a_{11} d\eta + \phi_\eta a_{12} d\eta\right)_{da} \Big) . \tag{3.11}$$

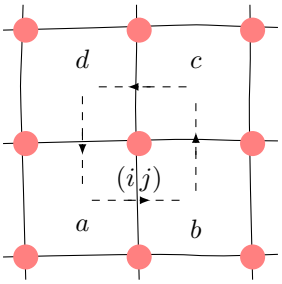


Figure 3.3: The control volume used for the finite-volume formulation of the Laplacian: using the divergence theorem, the Laplacian operator is rewritten in a finite-volume-like fashion using the rectangles *abcd* as the control volume.

Numerical boundary conditions

In addition to the previously mentioned boundary condition for the pressure (required due to the reformulation of the Navier-Stokes equations in pressure form), we have to consider the outflow boundary conditions for both the base flow and the eigenvalue computations, required because our domain is truncated at approximately 20% of the chord. While at the inflow and on the wing surface boundary condition can be defined based on physical arguments — velocities are given and the pressure is defined by the pressure boundary condition — the outflow boundary conditions remain undefined and challenging. As a consequence, a homogeneous Neumann boundary condition for the velocities and a Dirichlet boundary condition for the pressure have somewhat arbitrarily been tested. For the base flow computation, the pressure is taken as the inviscid solution at the outflow, which is expected to be a good approximation for high Reynolds number flow. For the perturbations, the pressure is required to be zero at outflow. In both cases, the Dirichlet boundary condition on the pressure gives rise to the development of rather sharp numerical boundary layer close to outflow, in particular in the case of the eigenvector results that, as we will see, are growing exponentially in the chordwise direction before reaching the outflow boundary. Nonetheless, as we will see and as has been previously suggested by Giannetti & Luchini [19], the outflow boundary condition have very little or no effect on the solution, provided that the outflow boundary is not too close to the attachment line — ten δ_{99} boundary layer thicknesses seem to be more than sufficient.

3.3 Validation

The implemented discretization of both the nonlinear and the linear problem needs to be validated, and a set of tests have to be performed in order to verify its correctness. Two tests are performed to validate the implemented discretization of the Navier-Stokes operator $\mathcal{R}_\Delta$ and an additional one to validate its linearization $\mathcal{L}_\Delta$.

First, the second-order accuracy of the scheme with respect to the mesh size is verified by substituting the analytical solution for the inviscid flow on grids with different mesh sizes h and computing the norm of the discrete residual as a function of h. The norm of the residual is expected to converge towards zero as h^2. Results are shown in Figure 3.4: the Euclidean $|\cdot|_2$ and maximum $|\cdot|_\infty$ norms of the residual are plotted using double-logarithmic axes as a function of the mesh size for the u-, v- and p-equations. After an initial transient, all norms show second-order convergence, represented by the black line. In this test, Dirichlet boundary conditions are applied along all boundaries and for all variables. The diffusion term in the momentum equations is indirectly validated since the same discretization is used for the Laplacian of the pressure. The w-momentum equation is the same as the u- and v-momentum equation, with the exception of the pressure spanwise derivative $\partial_z p$ which is zero for the base flow; it thus can be considered validated by the present test.

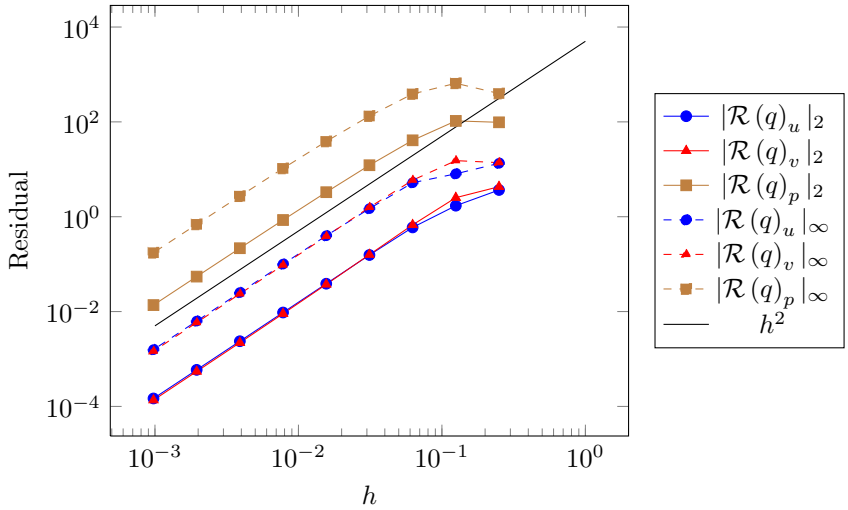


Figure 3.4: Norms of the numerical residual of the analytical solution for inviscid flow around a Joukowsky profile. The residual is computed by using the analytical solution on a grid of mesh size h and applying the corresponding discretization of the operator $\mathcal{R}_\Delta$. Both the Euclidean $|\cdot|_2$ and the maximum $|\cdot|_\infty$ norm converge to zero as h^2: the discretization is validated as second-order accurate.

As a second test the low Reynolds number flow around a circular cylinder is computed and compared with data from the literature and with a solution from the well established finite-element code FreeFem++ [52]. The main objective of this test is to validate the boundary condition implementation, and in particular the pressure boundary condition on the solid boundary. The domain used in this test is obtained by a conformal mapping of the form $x + iy = -e^{-i(\xi+i\eta)}$, where $\pi \leqslant \xi \leqslant \pi$ and $0 \leqslant \eta$. The maximum value of η determines the radial extent of the domain. This procedure is the same as the mapping from the rectangle to the circle in Figure 3.2. A representation of the generated grid is shown in Figure 3.5. The flow field is computed for $Re_D = 20, 40$ and 50, where $Re_D = U^*_\infty D^*/\nu^*$ is based on the free-stream velocity U^*_∞ and the cylinder diameter D^*. The grid used in the computation has 513 points in both the radial and azimuthal direction, and the ratio of the external boundary diameter to the solid boundary diameter is approximately 22. Symmetry is imposed across the *ad* and *bc* boundaries, so that only minimal modifications of the implementation that will be used in the computations of the attachment-line flow are required. The multigrid solver described in chapter 4 is used for the computation of the solution.

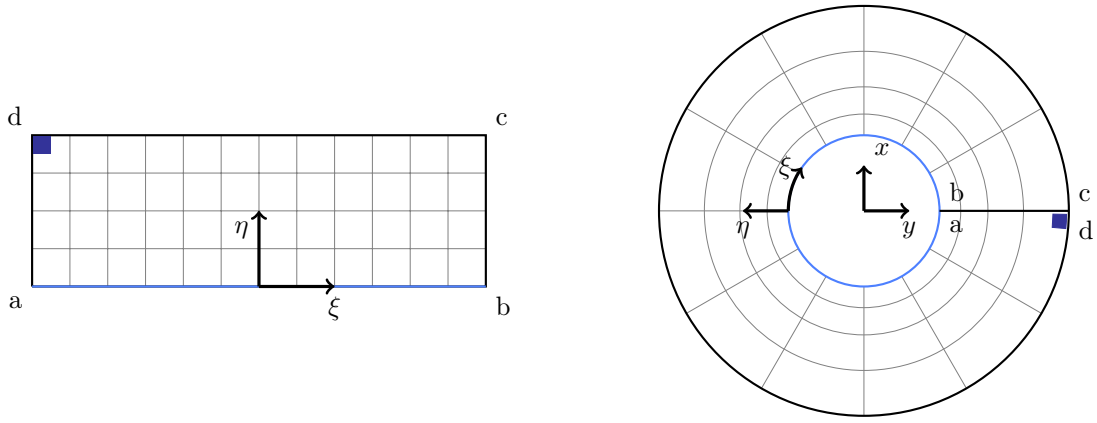


Figure 3.5: Conformal mapping from the rectangle to the circle. The mapping is in the form $x + iy = -e^{-i(\xi+i\eta)}$.

Table 3.2 compares the wake length and the forward and rear stagnation-point pressure from our solution with results previously computed by Fornberg [16] and Giannetti & Luchini [19] for Reynolds $Re = 20, 40$, as well as with results computed using FreeFem++.

Table 3.2: Wake bubble length and forward and rear stagnation-point pressure for different Reynolds numbers

	Wake length†			Pressure at forward stagnation point			Pressure at rear stagnation point		
Re_D*	20	40	50	20	40	50	20	40	50
Fornberg [16]	2.82	5.48	—	0.64	0.57	—	-0.27	-0.23	—
Giannetti [19]	2.84	5.48	—	—	—	—	—	—	—
FreeFem++	—	—	—	0.64	0.58	—	-0.27	-0.24	—
Current	2.82	5.49	6.82	0.63	0.57	0.56	-0.27	-0.24	-0.23

† in cylinder radii and measured from the center of the cylinder.
* The Reynolds number $Re_D = U_\infty D/\nu$ is based on the cylinder diameter D.

Additionally, Figure 3.6 compares the pressure distribution along the cylinder surface computed with the current discretization (continuous red and blue lines) and the one computed with FreeFem++ (black dashed lines) for the two Reynolds numbers $Re_D = 20, 40$. For both Reynolds numbers the pressure distribution is nearly indistinguishable over most of the cylinder surface. The small discrepancy at the forward stagnation point $\theta = 0$, visible also in the data in Table 3.2 is dependent on the domain size and diminishes as the domain size used in the FreeFem++ computation is increased.

The discretization of the linearized equation ($\mathcal{L}_\Delta$) is validated by comparing a finite-difference approximation of the Jacobian with the implemented one and verifying that the difference is on the order of roundoff errors. The finite-difference approximation is computed starting from the already validated nonlinear discretization. The tools implemented in the PETSc suite [5] are used for this step of the validation.

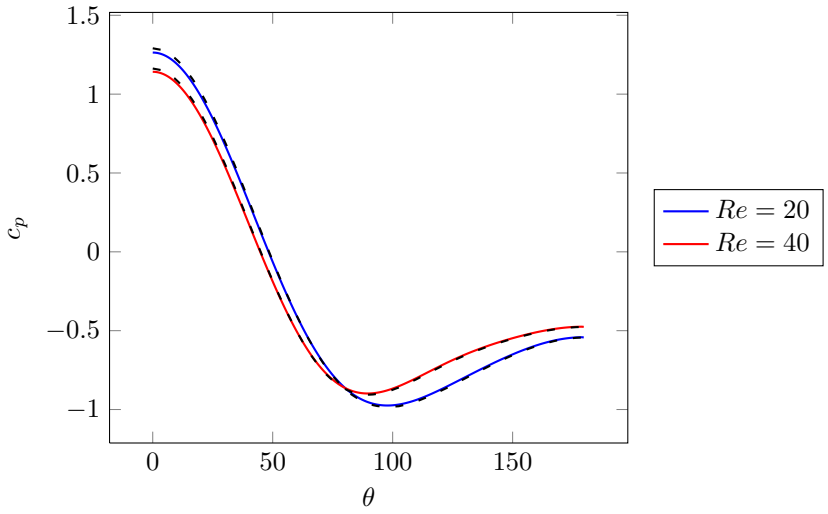


Figure 3.6: Pressure coefficient $c_p = \dfrac{p - p_\infty}{0.5\rho U_\infty^2}$ on the cylinder surface as a function of the angle. The forward stagnation point is at $\theta = 0$. Continuous lines represent data obtained from the current code, dashed black lines are data obtained with FreeFem++ [52].

3.4 Considerations on domain size and grid size

In order to get first approximations on the size of the computational problem that we are going to solve, we will proceed to estimate the mesh size that will be used in chapter 5. As already noted, the problem is characterized by two length scales. The larger scale, of $\mathcal{O}(1)$, is given by the length of the profile chord and provides the characteristic length on which the inviscid flow and the pressure field vary. The smaller scale is of size $\mathcal{O}\left(1/\sqrt{Re_C}\right)$ and determines the characteristic length of the boundary layer. At a chord based Reynolds number of $Re_C = 10^6$, the scale separation is thus of order 10^3. Our global approach requires to correctly represent both scales.

To correctly represent the inviscid scale, we select a domain extending roughly 1.5 chord lengths — or a more reassuring 100 leading edge radii — in the direction normal to the profile. The extent in the chordwise direction covers 20% of the chord, corresponding to an arc length of nearly 0.4 of the chord's length. The full domain is marked with a blue line in Figure 3.7, and its extent is deemed sufficient, taking into account the fact that the inviscid solution around the Joukowsky profile, contrary to a uniform flow, is used at the inflow boundary.

Within the boundary layer, we require the δ_{99} thickness to be discretized with nearly forty points and the mesh spacing in the chordwise direction to be nearly double the mesh spacing in the normal direction. Details of the grid close to the attachment line and the outflow boundary are shown in Figure 3.8. At first sight, we may expect to be able to choose a much larger mesh spacing in the chordwise direction as the boundary layer changes substantially faster in the normal direction than in the chordwise direction. We do not follow this approach for two reasons: (i) close to the attachment line, changes in both directions are of the same order of magnitude, and (ii) results from stability computations are expected to oscillate in the chordwise direction on a length scale similar to the boundary layer thickness.

The boundary layer δ_{99} thickness varies from $0.35 \cdot 10^{-3}$ at the attachment line to $1.5 \cdot 10^{-3}$ at the outflow, growing by a factor of four over the chordwise extent of the domain. If, for now, we consider an average chordwise mesh size of $10^{-3}/20$, then 8000 mesh points are required to cover the full chordwise extent. Grid stretching and adaptive grid refinement make any estimate of the number of grid points in

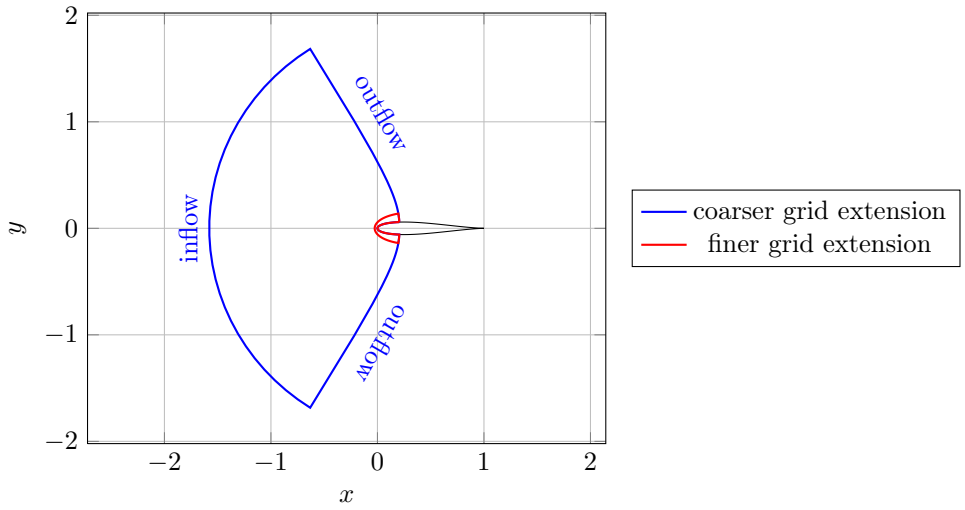


Figure 3.7: Domain for base flow computations. One level of adaptive mesh refinement is used, and the domain size is covered by two grids. The coarser of the two grids, whose extension is marked in blue, covers the entire domain and consists of 4097 points in the chordwise direction and 1025 points in the normal direction. Half of the points in the normal direction are contained within the red line. The finer mesh, whose extension is marked in red, has half the mesh size of the coarser grid and consists of 8193 points in the chordwise direction and 1025 points in the normal direction. The equivalent combined grid consists of 10.5 millions points. The boundary layer thickness at $Re_C = 10^6$ is too thin to be visualized.

the normal direction more involved, and it suffices to say that we will be using the equivalent of 1537 mesh points. The resulting problem then consists of more than 42 millions degrees of freedom when the four variables u, v, w, p are taken into account.

From a memory requirement point of view, this means nearly 350 MBytes to store a single vector and nearly 7 GBytes to store a sparse representation of the discretized linearized problem when real, double precision arithmetic is used.

High-performance solution algorithms are required to address the numerical problem. Considering a Newton solver, direct LU decomposition of the Jacobian matrix is not feasible. Even an iterative Krylov-subspace linear solver, coupled with ILU preconditioning with zero levels of fills, would require an additional 9 GBytes to store 30 Krylov vectors and an additional 7 GBytes to store the ILU decomposition, for a total of 23 GBytes without taking into account the working vectors.

In the next chapter we will introduce multigrid as a highly effective framework for computing discrete solutions of the Navier-Stokes problem. Starting from a simple Poisson equation we will proceed to the solution of the entire Navier-Stokes system in its steady-state form, thus avoiding the necessity of long-time time-stepping to compute the base flow. Multigrid demonstrates its efficiency both in terms of CPU time and memory — a few hours on a single processor and a few working vectors, respectively — required to converge towards the solution.

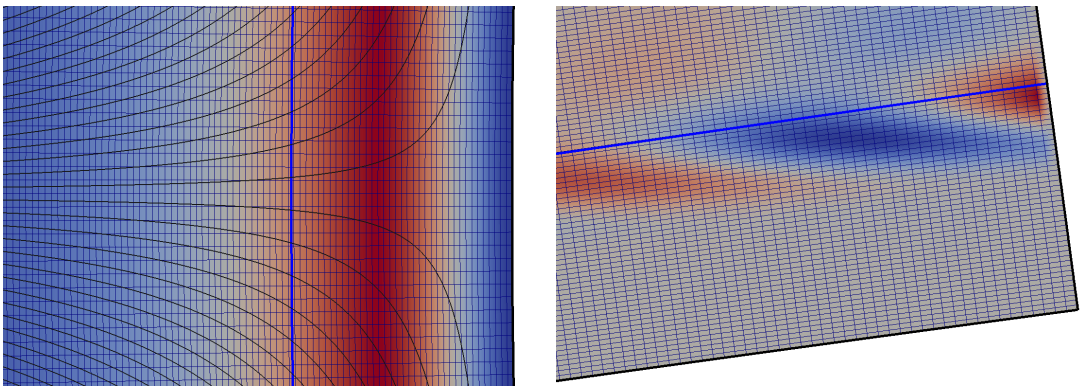


Figure 3.8: Zoom of the grid close to the attachment line (left) and close to the outflow boundary (right) of the domain represented in Figure 3.7. The grid is shown in thin blue lines. The typical structure of an eigenvector solution is represented in color and the δ_{99} boundary layer thickness with a blue, thick line. Streamlines in the x, y plane are shown in black in the attachment-line region — and close to outflow are parallel to the boundary. The two figures do not have the same scale: δ_{99} at the outflow is four time bigger than at the inflow.

Chapter 4

Multigrid

A good carpenter does not blame his tools

In this chapter the multigrid framework used to compute the discrete solution of the steady-state Navier-Stokes equations is presented. As has been briefly demonstrated at the end of the previous chapter, where it was stated that solving the Navier-Stokes equations becomes exceedingly expensive in terms of CPU time and memory requirements as the Reynolds number is increased: multigrid represents a far more efficient framework for performing this task.

First developed in the late 1970s by Achi Brandt [7] in order to address the solution of elliptic equations, multigrid is part of a class of iterative solution algorithms: the search for a solution to a given discrete problem is performed by iteratively computing a series of approximations starting from an initial guess, until a solution satisfying appropriate error criteria is found. This is in contrast to computing a solution by a direct approach like Gaussian elimination (or LU decomposition). Iterative solution methods include simple algorithms like Jacobi or Gauss-Seidel iterations as well as more complex ones like conjugate gradient, GMRES or multigrid itself [66]. The main advantages of iterative methods over direct methods are a more efficient use of memory and, in most cases, a reduced computational cost, allowing for the solution of significantly larger problems. In order to exemplify this, Table 4.1 (from [66]) presents the computational costs associated with the solution of the linear two-dimensional Poisson equation using various common algorithms: multigrid is clearly the most efficient solver for this simple case. As we will show later in this chapter, this is even more the case for more complex problems.

Table 4.1: Complexity of different solvers for the two-dimensional Poisson problem (from [66])

Method	# operations*
Gaussian elimination	$\mathcal{O}\left(N^2\right)$
Jacobi iteration	$\mathcal{O}\left(N^2 \log \varepsilon\right)$
Gauss-Seidel iteration	$\mathcal{O}\left(N^2 \log \varepsilon\right)$
Successive overrelaxation (SOR)	$\mathcal{O}\left(N^{3/2} \log \varepsilon\right)$
Conjugate gradient (CG)	$\mathcal{O}\left(N^{3/2} \log \varepsilon\right)$
Fast Fourier Transform (FFT)	$\mathcal{O}\left(N \log N\right)$
Multigrid (iterative)	$\mathcal{O}\left(N \log \varepsilon\right)$
Multigrid (FMG)	$\mathcal{O}\left(N\right)$

* N denotes the total number of unknowns in the discretized problem. The $\log \varepsilon$ term reflects the assumption that the accuracy of the solution is in the range of the discretization accuracy.

A generic iterative solver can be approached from two points of view. The first is what we can call a

"global" point of view, where the solution process is seen as the iterative application of a given operator M to an initial guess q^0, until a satisfactory solution q^m, whose error is sufficiently small, is found. The second is what we can call a "local" point of view; it is used to describe the internal workings of the operator M (or parts of it). Analysis of the operator M from a local point of view will provide us with in-depth informations on the behavior of Fourier components — which are also the formal eigenfunctions of M — under the repeated application of the operator M.

In section 4.1 we introduce a formal description of a generic operator-splitting-based iterative process. It is well known that the asymptotic convergence rate of any iterative process is given by the spectral radius of the operator M. In section 4.2 the main ingredients of a multigrid solver are introduced: these are (i) a hierarchy of grids G_h characterized by different mesh sizes h; (ii) a relaxation method, which is an iterative solver by itself, whose goal is to reduce the high wavenumber (short wavelength) components of the error and (iii) an interpolation and a restriction operator, whose role is to transfer informations between grids. The multigrid algorithm presented in this thesis is the Full Approximation Scheme (FAS), perhaps the lesser known, but more powerful, of the two multigrid algorithms — the other being the well-known Correction Scheme (CS). FAS has been chosen as the algorithm used in the code developed for this work because of its capability of handling nonlinear equations and of handling adaptive grid refinement. The main idea behind FAS is to drive the discrete equations on each grid (except the finest) with a forcing term τ_h^H, called the defect correction, in such a manner that the solutions on the coarse grids are identical to the one on the finest. Its relation to the Correction Scheme (CS) will be clarified in section 4.5.

In section 4.3 the relaxation method is introduced and analyzed by using Local Fourier Analysis (LFA), a fundamental tool in the context of multigrid analysis. Relaxation being one of the more delicate components of multigrid, we devote a significant portion of this chapter to it.

As a first step we consider one of the most commonly encountered problems in physics, the Poisson problem. Gauss-Seidel iteration — or one of its variants — is often used to solve the Poisson equation, mainly due to its simple implementation and memory efficiency. Despite these advantages, a quick look at Table 4.1 seems to suggest that Gauss-Seidel iteration is among the algorithms requiring the largest computational effort; however, this assessment is unfair. Looking at it in more detail, the description of the Gauss-Seidel iteration from a local point of view shows that it is very efficient in reducing the high-wavenumber components of the error $e^m = \bar{q} - q^m$, with $\bar{q}$ as the exact solution of the discrete problem, but rather slow at reducing the low-wavenumber components. Together with the observation that the term "high-wavenumber" is dependent on the mesh size h, this statement gives a first hint of the underlying idea of the multigrid algorithm: by using a discretization of the same problem on grids of different size, *all* error components can be reduced in an efficient way.

The performance of a Gauss-Seidel iteration — or, in the multigrid context, Gauss-Seidel relaxation — as well as the performance of any other relaxation process can be described by its amplification factor $\rho(\boldsymbol{\theta})$, measuring the amount of amplitude decay/growth over one relaxation sweep in a Fourier component of wavenumber $\boldsymbol{\theta} = \{\theta_x, \theta_y\}$. The smaller the amplification factor, the faster the error amplitude is reduced. An amplification factor greater than one denotes divergence of the iterative process. The relation to the spectral radius is immediate: the spectral radius is defined as the maximum of the amplification factor over all wavenumbers $\boldsymbol{\theta}$ representable on the grid, i.e.,

$$\bar{\rho} = \max_{\boldsymbol{\theta}} \rho(\boldsymbol{\theta}) .$$

As a second step, we slightly increase the difficulty of the problem by considering the scalar, linear

convection-diffusion equation. Local Fourier Analysis (LFA) suggests that the amplification factor of the Gauss-Seidel relaxation changes with respect to the Poisson equation case. Simple considerations on how and why the Gauss-Seidel relaxation properties — in particular its amplification factor $\rho(\boldsymbol{\theta})$ — are modified by the addition of extra terms provides valuable information on how to avoid a deterioration of the multigrid performances. We remark, for example, that a purely convective process can be solved very efficiently by downstream marching.

As a third step, we return to the Poisson equation and introduce anisotropy to the problem. The origin of such an anisotropy can be varied and can include physical characteristics of the problem or, more interestingly in our case, grid stretching. The convergence properties of the standard pointwise Gauss-Seidel iteration can be noticeably degraded by the presence of such an anisotropy. An efficient remedy is identified in a variation of the Gauss-Seidel algorithm, known as the linewise Gauss-Seidel iteration, where unknowns located on the same mesh line are solved collectively.

As a fourth and final step we consider the relaxation of a system of equations comprising both Poisson and convective-diffusion-reaction equations, i.e., a system like the pressure form of the linearized Navier-Stokes equations ($\mathcal{L}_\Delta$) described in section 3.1. A complete LFA analysis of the system of equations is not necessary, and we will only show how the scalar analysis results can be used instead. The task of devising a proper relaxation strategy for this system can readily be reduced to the adaptation of the relaxation strategies used for the Poisson and the convection-diffusion equation.

Once we have the tools to analyze the relaxation process, i.e., the one-grid process that smoothes the high-wavenumber components of the error, we can proceed to the analysis of the entire multigrid process. Two different schemes are available.

In section 4.4 we consider the widely known Correction Scheme (CS) used for the solution of linear equations, like the Poisson equation for the pressure in the fractional-step techniques. As noted, different grids characterized by different mesh spacings are employed to obtain a small amplification factor for all error components. In the CS scheme, the solution is represented only on the finest grid: coarser grids are used to compute corrections to the information stored on the next finer grid. This approach limits the applicability of CS to linear problems. At the end of this section possible multigrid strategies are presented, such as the V-cycle, the FMG algorithm and the FV-cycle.

In section 4.5 we introduce the perhaps lesser-known Full Approximation Scheme (FAS). FAS is used in the multigrid code developed in this work and can be obtained by taking a dual approach to the Correction Scheme (CS). FAS has two main advantages over CS: (i) the ability of solving nonlinear equations without the need of an outer Newton iteration, and (ii) a natural approach to the implementation of adaptive grid refinement strategies. At the basis of the FAS scheme lies the idea of representing the full solution, instead of corrections, on all grids; the discrete equations on each grid (except the finest) are forced such that the solution on any grid corresponds to the solution on the finest grid. Loosely speaking, this forcing can be related to the one used in deferred correction methods. Adaptive grid refinement will be considered at the end of this section.

Despite their different approach, CS and FAS share nearly all ingredients of a general multigrid strategy: a relaxation process, an interpolation strategy denoted by the operator I, a restriction strategy denoted by the operator R and a grid traversal protocol, the latter including the number of relaxation iterations (or sweeps) per grid and the order in which the different grids are processed.

Some real-life examples will be presented at the end of this chapter.

For the development of the present multigrid code, various data structures and routines from the PETSc library [5, 4, 6] have been used, in particular, relating to vector and matrix representations and for accessing direct, sparse solver for linear and nonlinear problems. The multigrid part has been

appropriately adapted to incorporate these features. The designed code is more general than the PETSc-internal multigrid code since it allows for adaptive mesh refinement and more sophisticated relaxation algorithms.

4.1 A generic iterative solver

An iterative solver can be approached in two different ways. The first is what we can call a "global" point of view, where the solution process will be considered as an iterative application of a given operator M to an initial guess q^0, until a sufficiently converged solution q^m has been found. The second point of view can be referred to as the "local" point of view; it is based on a description of the internal details of the operator M.

We start by taking the global point of view and consider a generic linear problem given by the equation

$$Aq = f \tag{4.1}$$

where A is an invertible matrix. A direct solution of the equation obtained by inverting the matrix A can be formally written as

$$q = A^{-1} f,$$

but the inversion of the matrix A (or the computation of its LU decomposition) is often too costly in terms of memory and CPU time to be practically feasible.

An alternative approach is to consider an iterative scheme. In this case, we split the matrix A in two parts, A^+ and A^-, such that

$$A = A^+ + A^-. \tag{4.2}$$

We can then define an iterative algorithm as

$$A^+ q^{m+1} = -A^- q^m + f \qquad \Longrightarrow \qquad q^{m+1} = M q^m + s \tag{4.3}$$

where $M = (A^+)^{-1} A$ and $s = (A^+)^{-1} f$, while q^{m+1} and q^m are approximate solutions at the iteration $m+1$ and m of the iterative process.

Starting from an initial guess q^0, at each iteration of the solution process the operator $M = (A^+)^{-1} A^-$ is applied to the available approximate solution q^m in order to reduce the error $e^m = \bar{q} - q^m$ with respect to the exact solution of the discrete problem $\bar{q}$.

The design of the splitting $A = A^+ + A^-$ — or equivalently of the operator M — defines the properties of the iterative algorithm. Of particular interest in the design process is to characterize how fast the error e^m goes to zero. An equation for the evolution of e^m at each iteration can be readily obtained by substitution of $q^m = \bar{q} - e^m$ and $q^{m+1} = \bar{q} - e^{m+1}$ in the iterative process (4.3), thus obtaining

$$\hat{A} e^{m+1} = R e^m \qquad \Longrightarrow \qquad e^{m+1} = M e^m. \tag{4.4}$$

Expansion of the error e^m into eigenfunctions of the operator M helps in clarifying the error dynamics by showing that the convergence rate of each eigenfunction is given by the corresponding eigenvalue. We characterize each eigenfunction by its amplitude ε and its shape ψ such that an appropriately chosen

norm of ψ is one. The error can then be written as

$$e^m = \sum_i \varepsilon_i^m \psi_i \tag{4.5}$$

where i varies over all eigenfunctions. For each eigenfunction $\varepsilon_i \psi_i$ there is an associated eigenvalue λ_i, and the effect of applying the operator M on the amplitude of the eigenfunction is given by the product of the eigenvalue and the amplitude itself, so that

$$\varepsilon_i^{m+1} = \lambda_i \varepsilon^m. \tag{4.6}$$

If the absolute value of all eigenvalues is less than one, the iterative process converges, and its asymptotic convergence is given by the largest eigenvalue (or spectral radius) $\tilde{\rho}$ of the operator (M) defined as

$$\tilde{\rho}(M) = \max\left\{|\lambda| : \lambda \text{ eigenvalue of } M\right\}, \tag{4.7}$$

such that asymptotically we have $||e^{m+1}|| \leq \tilde{\rho}(M)\,||e^m||$. The eigenfunction corresponding to the spectral radius is the slowest-decaying eigenfunction in the spectrum.

As the spectral radius approaches one, the error reduction process slows significantly, and more iterations are required to converge to the solution. As we will see shortly, this is the case for "smooth" eigenfunctions when a pure Gauss-Seidel iteration procedure is employed. The role of multigrid is to improve this situation: to correctly design a multigrid algorithm we will have to consider not only the largest eigenvalue of the Gauss-Seidel iteration but also the relationship between the eigenvalues and the shape of the corresponding eigenvectors.

4.2 Ingredients of a multigrid solver

We use this section to present a brief introduction to the multigrid framework. It should be sufficient to understand the ideas underlying the multigrid algorithm used in this work. In the subsequent sections more details will be given on each building block of a generic multigrid algorithm, and the choices made in this work will be justified. For the reader interested in an in-depth understanding of the multigrid framework in general and its application to fluid dynamics in particular, Brandt's *Multigrid Techniques: 1984 Guide with Applications to Fluid Dynamics* [8], the NASA technical reports by Diskin et al. [14] and Swanson [64], and the 2003 Annual Review by Thomas et al. [65] are recommended.

The design of a multigrid solver has been visualized in Figure 4.1. A hierarchy of grids is defined starting from a coarse grid which is refined by a factor of two (in each direction) at every successive grid level. Thus, the mesh size of each finer level is half the mesh size of the previous one. The governing equations are then discretized on each grid yielding a series of discrete operators $L_h^k(\cdot)$ $(k = 1 \ldots n)$ where n is the total number of grids employed. On a given grid the solution is *relaxed* to reduce the high-wavenumber components of the error

$$e_h = \tilde{q}_h - q_h \tag{4.8}$$

where, as we will see in a moment, high wavenumbers are to be linked to the particular mesh size [66, chapter 4]. The solution is then *restricted* onto a coarse grid of mesh size $H = 2h$, where low wavenumbers of the finer grid are relaxed. This operation is applied recursively until the coarsest grid is reached; on this final grid the equations are solved. Corrections for the solution on the finer grids are then computed

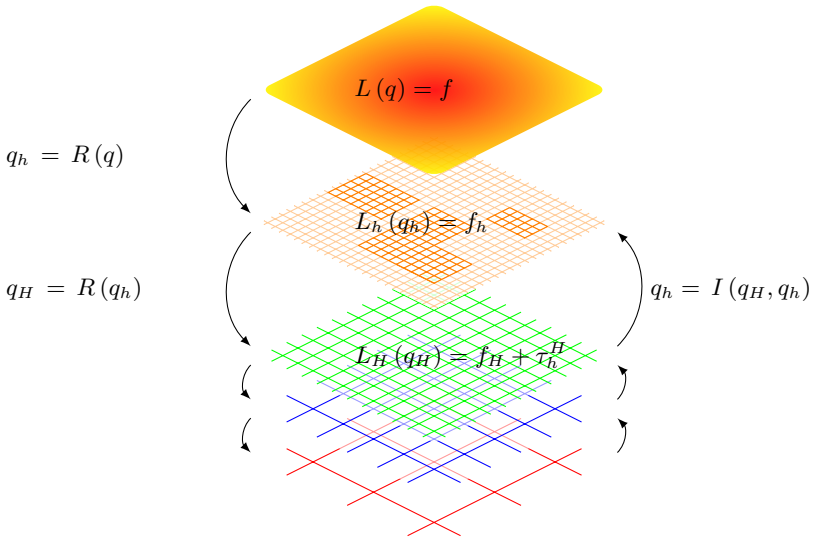


Figure 4.1: The grid hierarchy used in the multigrid process. The surface on the top represents the (unknown) analytical solution of the differential problem. R and I are the restriction and interpolation operators, respectively, and τ is the defect correction, a forcing term for the coarse grid equation with the role of equating the solution on all grids to the one on the finest grid. On each grid, a finite-difference problem L_h is defined by discretizing the continuous equations. If the Full Approximation Scheme (FAS) algorithm, that will be described in section 4.5, is used, the grids do not need to be coextensive, and the finer grids can cover only part of the coarser grids. This latter option allows the introduction of adaptive grid refinement.

and *interpolated* back. The restriction and interpolation operators are represented in Figure 4.1 by the symbols $\downarrow$ and $\uparrow$, respectively.

To give a more precise meaning to the terms *high* and *low* wavenumber, we consider a finer and a coarser grid of mesh size h and $H = 2h$, respectively, and a sinusoid $\sin(kx)$ with k as the wavenumber. The highest wavenumber that can be represented on each grid is $k = \pi/h$ and $k = \pi/H = \pi/(2h)$ for the fine and coarse grid, respectively. We can then determine all wavenumbers that are representable on the fine grid as

$$0 \leq \theta_{low} \leq \frac{\pi}{2h} < \theta_{high} \leq \frac{\pi}{h} \tag{4.9}$$

where sinusoids with wavenumber θ_{low} are correctly represented on both grids while sinusoids with wavenumbers θ_{high} are represented only on the fine grid (and aliased on the coarse one).

An essential component in the FAS multigrid algorithm is the fine-to-coarse defect correction τ_h^H. Its role is to force the coarse-grid equations in such a way that their solutions correspond to the fine-grid solution [8]. In this sense, on all grids, except the finest, a modified version of the discretized equations is relaxed (or solved):

$$L_H q_H = f_H + \tau_h^H \tag{4.10}$$

where

$$\tau_h^H = L_H(Rq_h) - R(L_h q_h). \tag{4.11}$$

How τ_h^H is obtained and more insights into its role will be given in section 4.5. At this point it is sufficient to state that τ_h^H takes into account the use of the coarse grid operator L_H instead of the fine grid operator L_h.

Once the FAS scheme has been chosen, two other ingredients need to be defined: a relaxation process

and the communication strategy between different grids.

Typically employed relaxation (or smoothing) algorithms for the Laplacian operator are the Jacobi and Gauss-Seidel iterations where each equation of the discretized system is solved sequentially. More complex relaxation strategies include line-Jacobi and line-Gauss-Seidel iterations — where the discrete equations corresponding to a line of points are solved simultaneously — and incomplete LU decomposition (ILU) iterations. For systems of equations the block Jacobi or block Gauss-Seidel iteration (or their corresponding block-line versions) can be employed to simultaneously solve all equations in the system at each mesh point. For equations that contain convective-type terms the direction in which the relaxation process is performed has been shown to be critically important [36]. Examples are the convection-diffusion equation and the momentum equations in the Navier-Stokes system. Further details will be given in section 4.3, and the reader is also referred to [36] and [66] for a full discussion of the smoothing properties of various relaxation schemes.

The goal of the communication strategy between two grids is (i) to provide an accurate representation of the finer-grid solution (or of the finer-grid error for the CS scheme) on the coarser grid and (ii) to interpolate the corrections computed on the coarser grid onto the finer grid. Two operators will be defined: a restriction operator and an interpolation operator, represented by $\downarrow$ and $\uparrow$, respectively, in Figure 4.1. The restriction operator can be chosen as a simple injection of the solution from the finer grid to the corresponding points of the coarser grid. Better performing restrictions are the half-weighted (HW) and full-weighted (FW) operators, which represent a weighted average of the five and nine mesh points of the fine grid surrounding a given mesh point on the coarser grid. The FW operator is used in the context of our code and is implemented as

$$q_H = \frac{1}{16}\Big(\begin{array}{ccccccc} 1q_{i-1,j+1} & + & 2q_{i,j+1} & + & 1q_{i+1,j+1} & + \\ 2q_{i-1,j} & + & 4q_{i,j} & + & 2q_{i+1,j} & + \\ 1q_{i-1,j-1} & + & 2q_{i,j-1} & + & 1q_{i+1,j-1} & \Big)\,. \end{array} \tag{4.12}$$

The interpolation operator can be a simple bilinear interpolation or a more complex second-order or third-order interpolation. A bilinear interpolation is often sufficient for an iterative multigrid method while a higher-order interpolation (usually at least of the order of the discretization scheme) is necessary for FMG multigrid. A third-order interpolation is used in this work.

It is important to note that in order to avoid introducing the interpolation errors of the whole solution, in the FAS scheme only the correction to the approximate solution computed at the previous iteration is interpolated from the coarser to the finer grid (something that is obvious in the correction scheme given that only the correction is represented on the coarser grid). In this way only the error related to the interpolation is introduced and can be easily eliminated with an additional iteration of the relaxation scheme [8]. The quantity to be interpolated is then $\left[q_H - \downarrow (q_h)\right]$ and the update to the previous iteration reads

$$q_h^{new} = q_h + \uparrow \left[q_H - \downarrow (q_h)\right]. \tag{4.13}$$

4.3 Relaxation

Despite the fact that we will use it as part of the multigrid solution process, Gauss-Seidel relaxation (or smoothing) can be considered as an iterative algorithm by itself and analyzed accordingly. We do so in order to uncover and overcome the problems associated with this most common — and extremely simple — iterative algorithm, and we will show that, when coupled with a multigrid approach, it is the most

efficient choice.

In the Gauss-Seidel relaxation, the discretized equation at each mesh point i, j, corresponding to a single line of the discretized operator, is solved locally in order to compute an update of the unknown $q_{i,j}$. This operation is repeated for all mesh points in the grids, using updated information when available and information from the previous iteration otherwise.

As we will see, the order in which the mesh points are addressed is not unimportant. The most simple and common ordering is given by the lexicographic (LEX) ordering and consists of traversing the grid with increasing values of the indices i, j. Another common ordering is the red-black (RB) ordering, where mesh points are organized like a checkerboard (or using more complicated patterns), and a sweep over the red mesh points is followed by a sweep over the black mesh points (yes, checkerboards are more often black and white...). We will also consider backward-lexicographic ordering, where one or both mesh directions are traversed with decreasing values of the indices i, j, and will see in which cases they can prove to be useful.

The Poisson equation and Local Fourier Analysis

We start by considering one of the most commonly encountered problems in physics, the scalar Poisson equation:

$$\Delta q = f \tag{4.14}$$

such that the operator A in equation (4.1) is now the Laplace operator. Our goal in this and the following sections is to perform and generalize the analysis outlined in section 4.1.

The first step is to define a standard five-point discretization L_h of the Laplacian operator on a two-dimensional grid with constant mesh size h. The discrete operator L_h can be represented locally — at every point i, j — by the discrete equation

$$L_h q_h = \frac{q_{i+1,j} - 2q_{i,j} + q_{i-1,j}}{h^2} + \frac{q_{i,j+1} - 2q_{i,j} + q_{i,j-1}}{h^2} = f_{i,j}. \tag{4.15}$$

An alternative notation for describing the discretization of an operator that will prove very convenient in the subsequent analysis is the stencil notation [66]. In such a notation, which provides a graphical representation of the stencil used in the discretization, the discrete Poisson equation (4.15) is written as

$$L_h q_h = \frac{1}{h^2} \begin{bmatrix} & 1 & \\ 1 & -4 & 1 \\ & 1 & \end{bmatrix}_h q_h(x, y) = f_h(x, y). \tag{4.16}$$

The application of the stencil L_h to the variable q_h is defined by

$$L_h q_h = [s_{\boldsymbol{\kappa}}]_h q_h = \sum_{\boldsymbol{\kappa}} s_{\boldsymbol{\kappa}} q_h(\mathbf{x} + \boldsymbol{\kappa}\mathbf{h}) \tag{4.17}$$

where $s_{\boldsymbol{\kappa}}$ is the set of coefficients of the discretization identified by the index sets $\boldsymbol{\kappa}$, and the summation is performed over the entire index sets $\boldsymbol{\kappa}$ belonging to the stencil. Taking the current case as an example, we have $s_{\boldsymbol{\kappa}} = -4/h^2$ for $\boldsymbol{\kappa} = (0, 0)$ and $s_{\boldsymbol{\kappa}} = 1/h^2$ for $\boldsymbol{\kappa} = (\pm 1, 0)$ and $\boldsymbol{\kappa} = (0, \pm 1)$.

We can now apply the analysis outlined in the previous section to the operator L_h: we implement the operator splitting corresponding to the Lexicographic Gauss-Seidel (GS-LEX) iteration, identify the iteration operator M and perform the modal analysis in order to identify the spectral radius $\tilde{\rho}$ and the

amplification factor ρ. This same analysis has been performed in multiple publications and in particular we refer to the 1982 paper by Kettler [36] and to the book by Trottenberg et al. [66]. Kettler [36] analyzes and summarizes results for a variety of model problems and relaxation methods, including point and line Gauss-Seidel, incomplete LU decomposition (ILU) and their variants. Some of his results will be used and extended in what follows.

The application of a GS-LEX iteration (or sweep) consists of successively solving each discrete equation (4.15) defined at every mesh point i, j on the grid, such that at iteration $m+1$ we update the unknown $q_{i,j}$ as

$$q_{i,j}^{m+1} = \frac{q_{i-1,j}^{m+1} + q_{i,j-1}^{m+1} + q_{i+1,j}^{m} + q_{i,j+1}^{m} + h^2 f}{4} \tag{4.18}$$

where the unknowns at the mesh points $i-1, j$ and $i, j-1$ are available at the current iteration $m+1$ while the others are at the previous iteration level m. This is a consequence of the lexicographic ordering of the sweep. From an operator splitting point of view, we can define the splitting as $L_h = L_h^+ + L_h^-$ which, in stencil notation, reads

$$L_h^+ = \frac{1}{h^2}\begin{bmatrix} & 0 & \\ 1 & -4 & 0 \\ & 1 & \end{bmatrix}_h \qquad L_h^- = \frac{1}{h^2}\begin{bmatrix} & 1 & \\ 0 & 0 & 1 \\ & 0 & \end{bmatrix}_h . \tag{4.19}$$

In a more common matrix notation, if the discrete unknowns are ordered in lexicographic order, L_h^+ and L_h^- would be the lower (including the main diagonal) and the upper triangular part of L_h, respectively.

The resulting equivalent of the operator splitting-based iterative process (4.3) is then

$$L_h^+ q^{m+1} = -L_h^- q^m + f \qquad \Longrightarrow \qquad q^{m+1} = -\left(L_h^+\right)^{-1} L_h^- q^m + \left(L_h^+\right)^{-1} f, \tag{4.20}$$

and the iteration operator M can be formally written as

$$M_h = -\left(L_h^+\right)^{-1} L_h^- . \tag{4.21}$$

We already know that the asymptotic convergence rate of the operator M_h is given by its spectral radius $\tilde{\rho}\left(M_h\right)$. We are now interested in gaining more insight into the properties of the operator M_h, and we will do that by analyzing the operator's eigenvalues and eigenfunctions. It is clear that the direct computation of the eigenvalues and eigenvectors of the operator M_h — or even the computation of the operator M_h itself — becomes unfeasible as the number of degrees of freedom of the discrete problem increase; fortunately, such a computation also provides far more information than is necessary.

A common, lighter and in the end more useful approach used in the analysis of relaxation processes in the multigrid framework is given by a Local Fourier Analysis (LFA). LFA provides a concise description of the local — i.e. at a given mesh point i, j — effect of one step of the relaxation process by computing an amplification factor

$$\rho\left(\boldsymbol{\theta}\right) = \varepsilon_{\boldsymbol{\theta}}^{m+1} / \varepsilon_{\boldsymbol{\theta}}^{m} \tag{4.22}$$

where $\varepsilon_{\boldsymbol{\theta}}^{m}$ is the amplitude of the error eigenfunction associated with a given wavenumber $\boldsymbol{\theta}$. In the case of a complex $\varepsilon_{\boldsymbol{\theta}}$, its magnitude has to be considered instead. A full description of LFA is provided by Brandt [7, 8] and reviewed by Trottenberg [66]. Here we will present the main ideas in order to explain its use.

Local Fourier Analysis (LFA) starts by considering a constant coefficient discretized operator L_h

defined on an infinite grid $\mathbf{G}_h$ of mesh spacing $\mathbf{h} = (h_1, h_2)$. The constant-coefficient and infinite-grid hypothesis is less limiting than it may appear: LFA is intended to provide local information, and most nonlinear, variable-coefficient operators can be linearized locally and replaced locally by an operator with constant coefficients. We anticipate here that there are two important cases where linearization is not meaningful: (i) at locations where the equation's coefficients vary too strongly or even discontinuously between mesh points and (ii) in the vicinity of boundaries.

An analytical expression for the eigenfunctions and eigenvalues can be obtained for any constant-coefficient operator defined on an infinite grid, starting with a Fourier transform of the error $e^m = \bar{q} - q^m$ where we recall that $\bar{q}$ is the exact solution to the discrete problem and q^m is its computed approximation at iteration m. In two dimensions, each Fourier component is described by the grid function ψ_h, which is a function of the wavenumber $\boldsymbol{\theta} = (\theta_1, \theta_2)$, of the position $\mathbf{x} = (x_1, x_2)$ and of the mesh spacing $\mathbf{h} = (h_1, h_2)$ according to

$$\psi_h(\boldsymbol{\theta}, \mathbf{x}) = \varepsilon_{\boldsymbol{\theta}} e^{i\boldsymbol{\theta}\mathbf{x}/\mathbf{h}} = \varepsilon_{\boldsymbol{\theta}} e^{i\theta_1 x_1/h_1} e^{i\theta_2 x_2/h_2}. \tag{4.23}$$

As we noted when defining high and low wavenumbers in section 4.2, the maximum wavenumber we can represent on a grid of mesh size h is π/h, so that for our analysis it suffices to consider the interval $-\pi \leq \boldsymbol{\theta} < \pi$. On this interval all grid functions are linearly independent and represent the eigenfunctions of any constant-coefficient discrete operator.

The complete spectrum can now be easily obtained by applying the discrete operator to the grid function ψ_h. Taking as an example the operator L_h and starting from the definition given in equation (4.17), which, we recall, arises naturally from the stencil notation in equation (4.16), the expression $L_h\psi_h$ reads

$$L_h\psi_h = [s_{\boldsymbol{\kappa}}]\,\psi_h = \left(\sum_{\boldsymbol{\kappa}} s_{\boldsymbol{\kappa}} e^{i\boldsymbol{\theta}\cdot\boldsymbol{\kappa}}\right) \varepsilon_{\boldsymbol{\theta}} e^{i\boldsymbol{\theta}\mathbf{x}/\mathbf{h}} \tag{4.24}$$

where the term between brackets represents the symbol of the operator

$$\tilde{L}_h(\boldsymbol{\theta}) = \sum_{\boldsymbol{\kappa}} s_{\boldsymbol{\kappa}} e^{i\boldsymbol{\theta}\cdot\boldsymbol{\kappa}}. \tag{4.25}$$

We recall that $s_{\boldsymbol{\kappa}}$ is the ensemble of coefficients of the discretization identified by the index sets $\boldsymbol{\kappa}$ and the summation is performed over all index sets $\boldsymbol{\kappa}$ belonging to the stencil as we have seen in (4.16) and (4.17). The symbol of the operator defines the location of the operator's spectrum in the complex plane as a function of the wavenumber $\boldsymbol{\theta}$, and in this sense it can be considered as the eigenvalue distribution of the operator itself: we have now an eigenvalue $\tilde{L}_h$ and an eigenfuction ψ_h associated with every wavenumber $\boldsymbol{\theta}$. In the case of our two-dimensional five-point discretization (4.16) of the Laplacian the symbol/eigenvalue reads

$$\tilde{L}_h(\boldsymbol{\theta}) = \frac{1}{h^2}\left(e^{-i\theta_1} + e^{-i\theta_2} + e^{i\theta_1} + e^{i\theta_2} - 4\right) = \frac{2}{h^2}\left(\cos\theta_1 + \cos\theta_2 - 2\right). \tag{4.26}$$

We can now consider the symbols of the two operators L_h^+ and L_h^- corresponding to the splitting of the operator L_h and compute the symbol for the iteration operator $M_h = -(L_h)^{-1} L_h^-$. With this information available, we can determine how the error e_h^m evolves during the iterative process. For Lexicographic Gauss-Seidel relaxation, whose splitting is defined in stencil notation in (4.19), application

of the discrete operator to the grid function results in

$$\tilde{L}_h^+(\boldsymbol{\theta}) = \frac{1}{h^2}\left(e^{-i\theta_1} + e^{-i\theta_2} - 4\right) \tag{4.27a}$$

$$\tilde{L}_h^-(\boldsymbol{\theta}) = \frac{1}{h^2}\left(e^{i\theta_1} + e^{i\theta_2}\right), \tag{4.27b}$$

and the symbol for the iteration can be easily computed as

$$\tilde{M}_h(\boldsymbol{\theta}) = \frac{\tilde{L}_h^-}{\tilde{L}_h^+} = \frac{e^{i\theta_1} + e^{i\theta_2}}{e^{-i\theta_1} + e^{-i\theta_2} - 4}. \tag{4.28}$$

A clear description of the effect of the iteration can be obtained by recalling that the error behaves as $e^{m+1} = M_h e^m$ — see (4.4) in section 4.1). Once the eigenvalues and eigenvectors of the operator M_h have been computed, we can rewrite the same expression as

$$\varepsilon_{\boldsymbol{\theta}}^{m+1}\, e^{i\boldsymbol{\theta}x/h} = \tilde{M}_h\, \varepsilon_{\boldsymbol{\theta}}^{m}\, e^{i\boldsymbol{\theta}x/h} \qquad \Longrightarrow \qquad \rho = \left|\frac{\varepsilon_{\boldsymbol{\theta}}^{m+1}}{\varepsilon_{\boldsymbol{\theta}}^{m}}\right| = \left|\tilde{M}_h\right| \tag{4.29}$$

where all quantities depend on the wavenumber $\boldsymbol{\theta}$, and $|\cdot|$ denotes the magnitude of a complex value.

Figure 4.2 shows contour levels of the amplification factor $\rho(\boldsymbol{\theta})$ for the present case of Lexicographic Gauss-Seidel iterations applied to the five-point discretization of the Laplacian operator. It can clearly be seen that GS-LEX has very good amplification factors $\rho \ll 1$ for error components of a high wavenumber (marked with a gray background), but that the amplification factor approaches one as the wavenumber tends to zero. A direct consequence of this observation is that most of the $\mathcal{O}\left(N^2 \log \varepsilon\right)$ computational cost

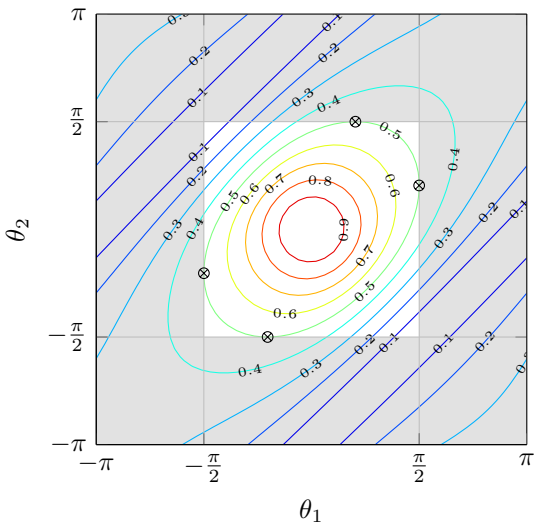


Figure 4.2: Contour plot of the amplification factor $\rho(\boldsymbol{\theta}) = |\varepsilon_{\boldsymbol{\theta}}^{m+1}/\varepsilon_{\boldsymbol{\theta}}^{m}| = |\tilde{M}_h|$ for a single sweep of Lexicographic Gauss-Seidel (GS-LEX) relaxation applied to the five-point discretization of the Laplacian operator, as a function of the θ_1 and θ_2 wavenumbers. The corresponding symbol is given by (4.28). The region of high wavenumbers which cannot be represented on a coarser grid of mesh size $H = 2h$, as defined by (4.9), is indicated by a gray background. The spectral radius $\bar{\rho}$ tends to one for low wavenumbers while the maximum amplification factor in the high wavenumber range is 0.5 and corresponds to the four wavenumbers marked with crossed circles.

associated with Gauss-Seidel iterations is spent on the reduction of low-wavenumber error components.

In contrast, the amplification factor for high wavenumbers is at most 0.5, which is a good smoothing factor: three Gauss-Seidel sweeps would reduce the amplitude of the high-wavenumber error component by a factor of $0.5^3 = 0.125$, or nearly an order of magnitude. We note here that the amplification factor for red-black Gauss-Seidel relaxation, at 0.25 per sweep [66], is considerably better in the case of the Poisson equation, but this is not true in the case of the convection-diffusion equation we will analyze in the next section; for this reason, red-black Gauss-Seidel relaxation is not considered in this work.

An additional remark on Figure 4.2 is that the amplification factor is not independent of the wavevector direction. This anisotropy is associated with the direction used during the sweep, i.e. the order in which the grid points are treated: the splitting used so far implies a sweep starting from the lower left (South-West) corner and ending in the upper right (North-East) corner. We can also consider the case when the sweep starts in the bottom right (SE) corner and ends in the top left (NW) corner. The corresponding amplification factor is displayed on the left of Figure 4.3: contours are mirrored about the vertical axis $\theta_1 = 0$ with respect to the previous case. The effect of alternating sweeps, the first starting in the SW corner, the second in the SE corner, are shown on the right of Figure 4.3: the isotropy is restored. The low

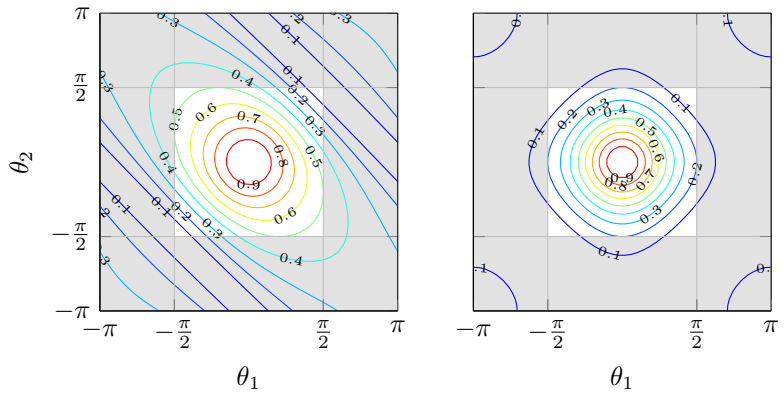


Figure 4.3: Same problem as Figure 4.2, but the ordering of the mesh points in the sweep is changed. For the left figure, the sweep proceeds towards decreasing i: the amplification factor is mirrored about the vertical axis. For the right figure, two sweeps are performed, the first for increasing i and the second for decreasing i. Better reduction factors are obtained (but at twice the computational cost), and the contours are symmetric with respect to both the horizontal and the vertical axis.

amplification factor of the high-wavenumber error components forms the basis of the multigrid design, but before moving onto this topic, we will consider three slightly more difficult problems: (i) the scalar, linear convection-diffusion equation, where the direction of the sweep will be much more important than in the Poisson case, (ii) the anisotropic Poisson equation and (iii) a system of equations similar to the linearized Navier-Stokes equations in pressure form.

The convection-diffusion equation

We will now consider a moderate increase in difficulty by addressing the scalar, linear convection-diffusion equation. Local analysis suggests that the amplification factor of the Gauss-Seidel relaxation changes compared to the Poisson equation case. Simple considerations on how and why the Gauss-Seidel relaxation's amplification factor $\rho(\boldsymbol{\theta})$, is modified will provide hints on how to avoid a deterioration of performance.

We start with the observation that a purely convective equation — i.e. without the Laplacian operator — can be solved very efficiently by a single pass of a downstream marching scheme.

We define the operator $\mathcal{Q}$ representing the two-dimensional, steady-state convection-diffusion equation as

$$\mathcal{Q}\, q = \mathbf{U} \cdot \nabla q - \nu \Delta q = f. \tag{4.30}$$

This is a notation similar to the one used in the previous chapters, but simplified in anticipation of investigating systems of equations in the next section.

A second-order discretization of the convection-diffusion equation can be obtained by using an upwinded, three-point discretization of the convective term and the standard five-point discretization of the Laplacian term. In stencil notation we can write

$$\left(\frac{1}{2h} \begin{bmatrix} & 0 & \\ & 0 & \\ U & -2U & 3\,(U+V) \quad 0 \quad 0 \\ & -2V & \\ & V & \end{bmatrix}_h - \frac{\nu}{h^2} \begin{bmatrix} & 0 & \\ & 1 & \\ 0 & 1 & -2 \quad 1 \quad 0 \\ & 1 & \\ & 0 & \end{bmatrix}_h \right) q_h\,(x,y) = f_h(x,y). \tag{4.31}$$

The first thing to be noted is that this discretization is valid only for $U, V \geq 0$. The stencil representing the convective term switches symmetrically about the central point and changes sign when the sign of the velocity components changes.

Analogous to what has been done for the Poisson equation, the operator L_h is split into a L_h^+ and a L_h^- part defining the Lexicographic Gauss-Seidel iteration:

$$L_h^+ = \frac{1}{2h} \begin{bmatrix} & 0 & \\ & 0 & \\ U & -2U & 3\,(U+V) \quad 0 \quad 0 \\ & -2V & \\ & V & \end{bmatrix}_h - \frac{\nu}{h^2} \begin{bmatrix} & 0 & \\ & 0 & \\ 0 & 1 & -4 \quad 0 \quad 0 \\ & 1 & \\ & 0 & \end{bmatrix}_h, \tag{4.32a}$$

$$L_h^- = \frac{1}{2h} \begin{bmatrix} & 0 & \\ & 0 & \\ 0 & 0 & 0 \quad 0 \quad 0 \\ & 0 & \\ & 0 & \end{bmatrix}_h - \frac{\nu}{h^2} \begin{bmatrix} & 0 & \\ & 1 & \\ 0 & 0 & 0 \quad 1 \quad 0 \\ & 0 & \\ & 0 & \end{bmatrix}_h. \tag{4.32b}$$

The symbols associated with the splitting can then be computed as

$$\tilde{L}_h^+ = -\frac{\nu}{h^2}\left(e^{-i\theta_1} + e^{-i\theta_2} - 4\right) + \frac{U}{2h}\left(e^{-2i\theta_1} - 2e^{-i\theta_1} + 3\right) + \frac{V}{2h}\left(e^{-2i\theta_2} - 2e^{-i\theta_2} + 3\right), \tag{4.33a}$$

$$\tilde{L}_h^- = -\frac{\nu}{h^2}\left(e^{i\theta_1} + e^{i\theta_2}\right), \tag{4.33b}$$

while the symbol for the iteration operator is

$$\tilde{M}_h(\boldsymbol{\theta}) = \frac{\tilde{L}_h^-}{\tilde{L}_h^+} = \frac{-\nu/h^2\left(e^{i\theta_1}+e^{i\theta_2}\right)}{-\nu/h^2\left(e^{-i\theta_1}+e^{-i\theta_2}-4\right)+U/(2h)\left(e^{-2i\theta_1}-2e^{-i\theta_1}+3\right)+V/(2h)\left(e^{-2i\theta_2}-2e^{-i\theta_2}+3\right)}. \tag{4.34}$$

In order to better understand this expression, it is convenient to multiply both the numerator and the denominator by $h/|\mathbf{U}|$. As a consequence, U and V in the denominator are replaced by $\sin\alpha$ and $\cos\alpha$, respectively, with α as the angle of the velocity vector with the horizontal axis. A Reynolds number based on the mesh size h can now be defined as $Re_h = |\mathbf{U}|h/\nu$ and the symbol can be rewritten as

$$\tilde{M}_h(\boldsymbol{\theta}) = \frac{Re_h^{-1}(D_n)}{-Re_h^{-1}(D_d)+\cos\alpha/2\ (C_1)+\sin\alpha/2\ (C_2)} \tag{4.35}$$

where D_d, D_n, C_1 and C_2 are the terms in brackets in equation (4.34). The first two are related to the diffusion term while the last two to the convection term in the x_1 and x_2 direction, respectively.

Two different asymptotic regimes can be considered depending on the value of the mesh-based Reynolds number Re_h: (i) for small $Re_h \ll 1$ the D_n and D_d terms dominate, and the situation of the pure Laplace operator treated in the previous section is recovered independently of the value Re_h, while (ii) for big $Re_h \gg 1$ the C_1 and C_2 terms are dominant in the denominator, and the amplification factor $\rho(\boldsymbol{\theta})$ tends to zero: indeed, for a purely convective equation, one single pass is enough to solve the equation exactly, provided that the sweep is performed in the downstream direction.

As previously done for the Laplacian operator, we plot in Figure 4.4 the amplification factor as a function of the wavenumbers. Contour levels of the amplification factor $\rho(\boldsymbol{\theta})$ are shown for Reynolds numbers $Re_h = 10^{-1}, 1, 10^1, 10^2$ and for a velocity $\mathbf{U}$ with an angle $\alpha = 45^\circ$ with respect to the horizontal axis of the grid.

Inspection of these contours plots confirms what has been established by considering the asymptotic regimes: an increase in Reynolds number Re_h reduces the amplification factor $\rho(\boldsymbol{\theta})$, increasing the convergence rate of the error at each Gauss-Seidel sweep. The error components characterized by a lower wavenumber, corresponding to smooth error components, are always the slower to converge, but the overall error reduction is greater than in the simple Poisson case.

We have already noted while investigating the relaxation of the Laplace operator that changes in the sweep direction does affect the convergence properties of the Gauss-Seidel relaxation. This feature is even more pronounced in the case of the convection-diffusion equation, due to the directionality of the convection operator associated with the discrete upwinding. As an example, we consider the effect of relaxation with the same splitting as (4.32) corresponding to a Lexicographic Gauss-Seidel sweep, but rather applied to a negative velocity field $U, V \leq 0$. Upwinding of the convective derivative results in a stencil representation of the convection-diffusion equation given by

$$\left(\frac{1}{2h}\begin{bmatrix} & & |V| & & \\ & & -2|V| & & \\ 0 & 0 & 3\left(|U|+|V|\right) & -2|U| & |U| \\ & & 0 & & \\ & & 0 & & \end{bmatrix}_h - \frac{\nu}{h^2}\begin{bmatrix} & & 0 & & \\ & & 1 & & \\ 0 & 1 & -2 & 1 & 0 \\ & & 1 & & \\ & & 0 & & \end{bmatrix}_h\right) q_h(x,y) = f_h(x,y) \tag{4.36}$$

where the only difference with respect to the positive velocity case (4.31) is in the convective term. The absolute value of the negative velocities U, V takes into account the change in sign of the stencil. The

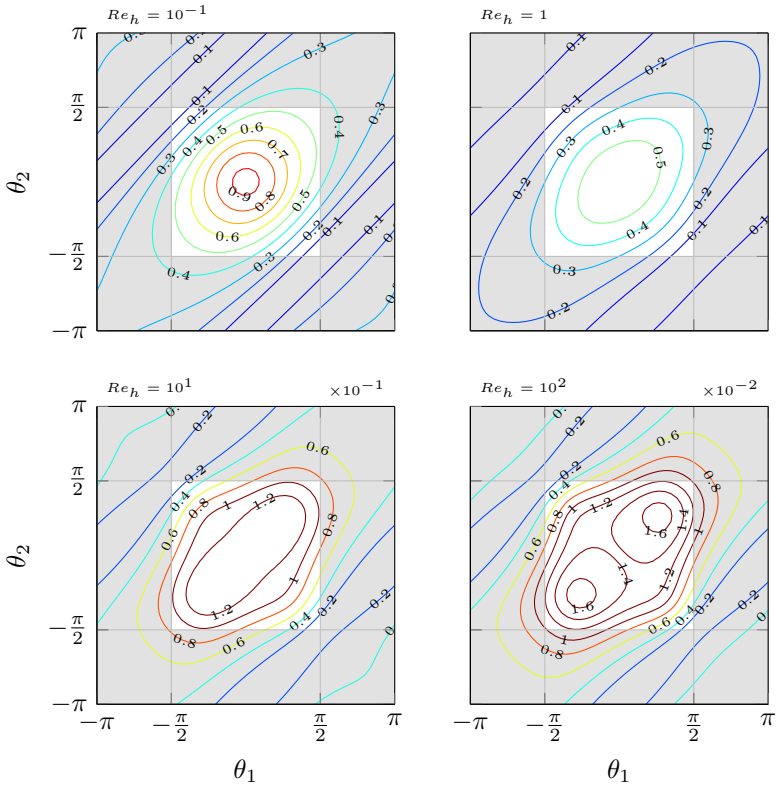


Figure 4.4: Contour plots of the amplification factor $\rho(\boldsymbol{\theta}) = |\varepsilon_{\boldsymbol{\theta}}^{m+1}/\varepsilon_{\boldsymbol{\theta}}^{m}| = |\tilde{M}_h|$ for the convection-diffusion equation. A constant flow field $U = V = 1/\sqrt{2}$ corresponding to $\alpha = 45°$ has been selected. Downstream marching is applied in all cases. Four mesh-based Reynolds number are shown: $Re_h = 10^{-1}$, 1, 10^1, 10^2. For the lowest $Re_h = 10^{-1}$ the contours are very close to the pure Poisson problem case, but the amplification factor improves when increasing the Reynolds number Re_h. Contours are rescaled in the two lower plots, as marked by the coefficient on the top right corner of these plots. The corresponding symbol is given by (4.34).

GS-LEX splitting considered above (4.32) now reads

$$L_h^+ = \frac{1}{2h}\begin{bmatrix} & & 0 & & \\ & & 0 & & \\ 0 & 0 & 3\left(|U|+|V|\right) & 0 & 0 \\ & & 0 & & \\ & & 0 & & \end{bmatrix}_h - \frac{\nu}{h^2}\begin{bmatrix} & & 0 & & \\ & & 0 & & \\ 0 & 1 & -4 & 0 & 0 \\ & & 1 & & \\ & & 0 & & \end{bmatrix}_h \tag{4.37a}$$

$$L_h^- = \frac{1}{2h}\begin{bmatrix} & & |V| & & \\ & & -2|V| & & \\ 0 & 0 & 0 & -2|U| & |U| \\ & & 0 & & \\ & & 0 & & \end{bmatrix}_h - \frac{\nu}{h^2}\begin{bmatrix} & & 0 & & \\ & & 1 & & \\ 0 & 0 & 0 & 1 & 0 \\ & & 0 & & \\ & & 0 & & \end{bmatrix}_h . \tag{4.37b}$$

The symbols associated with the splitting can then be computed as

$$\tilde{L}_h^+ = -\frac{\nu}{h^2}\left(e^{-i\theta_1} + e^{-i\theta_2} - 4\right) + 3\frac{|U|}{2h} + 3\frac{|V|}{2h}, \tag{4.38a}$$

$$\tilde{L}_h^- = -\frac{\nu}{h^2}\left(e^{i\theta_1} + e^{i\theta_2}\right) + \frac{|U|}{2h}\left(e^{2i\theta_1} - 2e^{i\theta_1}\right) + \frac{|V|}{2h}\left(e^{2i\theta_2} - 2e^{i\theta_2}\right) \tag{4.38b}$$

while the symbol for the iteration operator, using the mesh-based Reynolds number Re_h, is

$$\tilde{M}_h(\boldsymbol{\theta}) = \frac{\tilde{L}_h^-}{\tilde{L}_h^+} = \frac{-Re_h^{-1}\left(e^{i\theta_1} + e^{i\theta_2}\right) + |\cos\alpha|/2\left(e^{2i\theta_1} - 2e^{i\theta_1}\right) + |\sin\alpha|/2\left(e^{2i\theta_2} - 2e^{i\theta_2}\right)}{-Re_h^{-1}\left(e^{-i\theta_1} + e^{-i\theta_2} - 4\right) + 3|\cos\alpha|/2 + 3|\sin\alpha|/2}. \tag{4.39}$$

We can again consider the two asymptotic regimes: (i) for small Reynolds numbers $Re_h \ll 1$ the situation is unchanged, as can be expected given that there are no changes in the diffusion term; (ii) in contrast, for large Reynolds numbers $Re_h \gg 1$, the situation is different: the numerator does not tend to zero as Re_h^{-1}, and optimal convergence rates — characteristic of the case with positive velocity components — cannot be achieved. This effect has to be attributed to the fact that the relaxation process is sweeping in the wrong direction, or upstream. Information cannot be transferred in the optimal way because the stencils of the convective term use information from the old approximation instead of the new one, as is the case when downstream marching is employed. From another point of view, we can state that when downstream marching is employed, the convective term is solved implicitly, and the solution process is "equivalent" to a direct solution of the convective term: if the correct ordering of the unknowns is chosen, the matrix corresponding to the convective operator is already lower triangular and downstream marching corresponds to forward substitution.

Figure 4.5 shows the corresponding reduction factors for $\alpha = 180\,^\circ$ ($U = -1$) and $\alpha = 225\,^\circ$ ($U, V = -1/\sqrt{2}$). Amplification factors are very close to one for high wavenumbers indicating inefficiency of the relaxation scheme based on upstream marching for high Re_h grids.

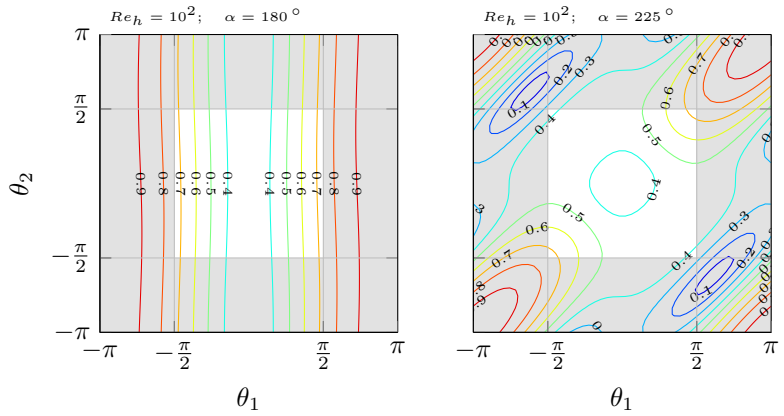


Figure 4.5: Contours of the amplification factor for the convection-diffusion problem, for a mesh-based Reynolds number of $Re_h = 10^2$. Upstream marching is used. Comparison with the bottom right plot in Figure 4.4, where downstream marching has been used, shows a strong degradation, by two orders of magnitude, in the amplification factor for both $U = -1, V = 0$ ($\alpha = -180°$, left plot) and for $U = V = -1/\sqrt{(2)}$ ($\alpha = -135°$, right plot). The sweep direction is paramount in regions of the computational domain with high mesh-based Reynolds numbers. The corresponding symbol is given by (4.39).

Line Gauss-Seidel relaxation

We now return our attention to the Poisson equation by considering the case of an anisotropic equation in the form

$$\varepsilon q_{xx} + q_{yy} = f. \tag{4.40}$$

If we consider the same splitting previously used for the isotropic Poisson problem and corresponding to a Lexicographic Gauss-Seidel iteration, the operators L_h^+ and L_h^- characterizing the relaxation scheme read

$$L_h^+ = \frac{1}{h^2}\begin{bmatrix} & 0 & \\ \varepsilon & -2(1+\varepsilon) & 0 \\ & 1 & \end{bmatrix}_h \qquad L_h^- = \frac{1}{h^2}\begin{bmatrix} & 1 & \\ 0 & 0 & \varepsilon \\ & 0 & \end{bmatrix}_h . \tag{4.41}$$

The anisotropy described by ε can represent a physical characteristic of the problem. Another more common possibility is to include ε after the discretization representing an anisotropy due to a grid stretching: in the present case we can set $h = h_2$ and $\varepsilon = (h_2/h_1)^2$.

The symbols associated with the pointwise Gauss-Seidel splitting are slightly modified with respect to the isotropic case by the introduction of ε. They read

$$\tilde{L}_h^+(\boldsymbol{\theta}) = \frac{1}{h^2}\left(\varepsilon e^{-i\theta_1} + e^{-i\theta_2} - 2(1+\varepsilon)\right), \tag{4.42a}$$

$$\tilde{L}_h^-(\boldsymbol{\theta}) = \frac{1}{h^2}\left(\varepsilon e^{i\theta_1} + e^{i\theta_2}\right), \tag{4.42b}$$

and the iteration operator M_h accordingly changes to

$$\tilde{M}_h(\boldsymbol{\theta}) = \frac{\varepsilon e^{i\theta_1} + e^{i\theta_2}}{\varepsilon e^{-i\theta_1} + e^{-i\theta_2} - 2(1+\varepsilon)}. \tag{4.43}$$

The associated amplification factors for $\varepsilon = 1, 1/4, 1/9, 1/16$ are depicted in Figure 4.6. From the figure it is evident that the contours of the amplification factor ρ spread in the θ_1 direction when reducing ε — that is, the reduction factors for high-wavenumber θ_1 deteriorate for increasing grid stretching.

A possible solution is to replace the pointwise relaxation of the Gauss-Seidel scheme — where the discrete equation at each mesh point i, j is updated independently — with a linewise Gauss-Seidel relaxation (LGS), where all unknowns on a mesh line are updated collectively. Line relaxation is somewhat more expensive than pointwise relaxation because a tridiagonal or pentadiagonal system has to be solved, but fast algorithms (with a computation cost that scales linearly with the number of unknowns) are readily available [3], and the increase in computation cost is easily manageable. These systems correspond to the one-dimensional discretization (along a mesh line) of the Laplacian or the convective-diffusion operator, respectively.

In stencil notation, LGS corresponds to the splitting

$$L_h^+ = \frac{1}{h^2}\begin{bmatrix} & 1 & \\ \varepsilon & -2(1+\varepsilon) & 0 \\ & 1 & \end{bmatrix}_h \qquad L_h^- = \frac{1}{h^2}\begin{bmatrix} & 0 & \\ 0 & 0 & \varepsilon \\ & 0 & \end{bmatrix}_h , \tag{4.44}$$

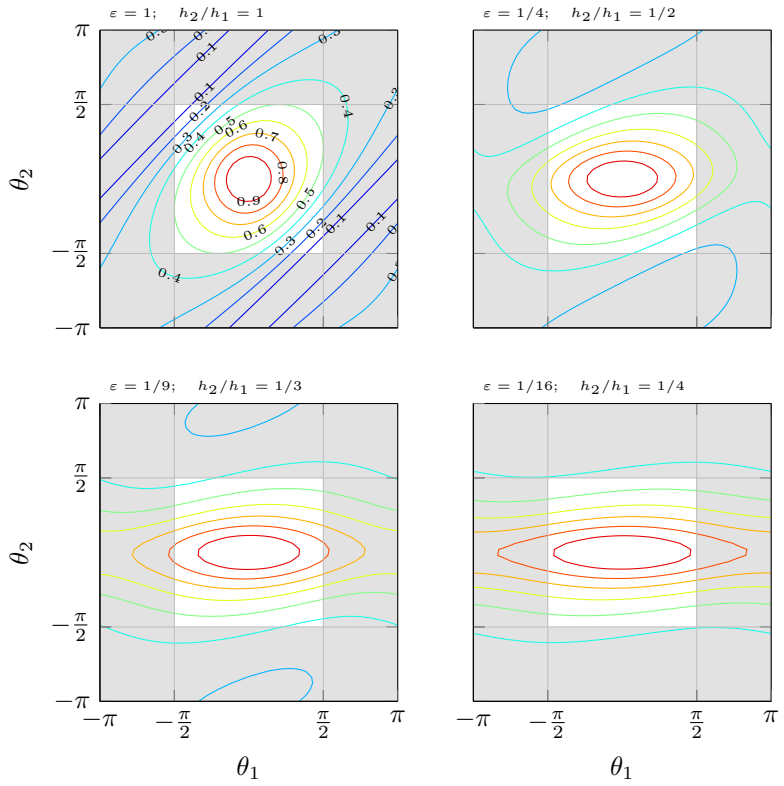


Figure 4.6: Contour plots of the amplification factor for the anisotropic Poisson equation. Increasing anisotropy leads to a spreading of the contours in the direction where the mesh size is largest. Accordingly, amplification factors increase for higher wavenumbers resulting in a decrease in the convergence rate. Values of the contours levels are marked only in the top left figure for clarity (as for the case of the isotropic Poisson problem), but all figures are color-coded in the same manner. The symbol for these plots is given by (4.43).

and the corresponding symbols are given by

$$\tilde{L}_h^+(\boldsymbol{\theta}) = \frac{1}{h^2}\left(\varepsilon e^{-i\theta_1} + e^{-i\theta_2} + e^{i\theta_2} - 2(1+\varepsilon)\right), \tag{4.45a}$$

$$\tilde{L}_h^-(\boldsymbol{\theta}) = \frac{1}{h^2}\left(\varepsilon e^{i\theta_1}\right). \tag{4.45b}$$

The symbol for the operator M_h can be computed accordingly as

$$\tilde{M}_h(\boldsymbol{\theta}) = \frac{\varepsilon e^{i\theta_1}}{\varepsilon e^{-i\theta_1} + e^{-i\theta_2} + e^{i\theta_2} - 2(1+\varepsilon)}. \tag{4.46}$$

Figure 4.7 shows the convergence factors for the same cases of Figure 4.6 but using linewise relaxation instead of pointwise relaxation. It is clear that the spread of the contours in the θ_1 direction, characteristic of pointwise relaxation, is no longer present. In spite, there appears a tightening of the contours in the θ_2 direction, with the consequence that high θ_2 wavenumbers are damped even faster than before — in essence, since the θ_2 direction is solved implicitly.

The Line Gauss-Seidel scheme also partially solves the problem of switching stencils in the advection discretization, as the direction of the stencil along the implicitly solved mesh line becomes irrelevant.

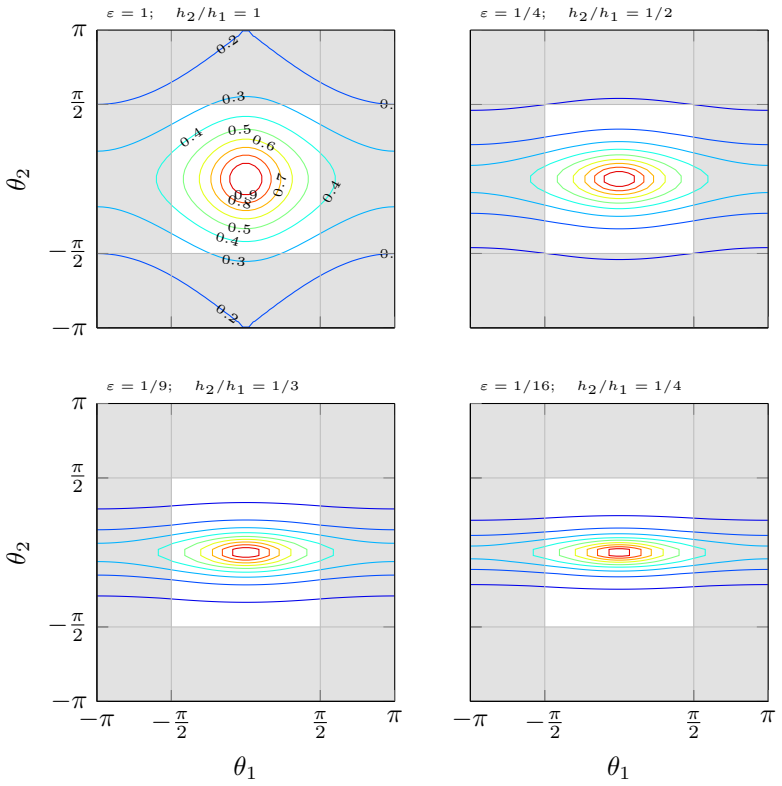


Figure 4.7: Same as Figure 4.6, but using linewise Gauss-Seidel relaxation instead of pointwise Gauss-Seidel relaxation. The spread of the contours in the horizontal direction, characteristic of pointwise Gauss-Seidel, is eliminated and, in its place, there is a tightening of the contours in the vertical direction, corresponding to better (smaller) amplification factors. Linewise Gauss-Seidel relaxation is somewhat more expensive than its pointwise equivalent as it requires the implicit solution of tri- or pentadiagonal systems: the reduction in amplification factor is not worth the additional computation cost in the isotropic case, but becomes essential for the anisotropic case. The symbol is given by (4.46).

It is important to remark that we have considered a *local* deformation, i.e., an anisotropy which is a characteristic of a single mesh point. This analysis does not cover the case of conformal mapping, where the stretched grid is still locally isotropic if an originally isotropic grid is used; this is the case for the grids showed in Figure 3.2. In fact, Local Fourier Analysis (LFA) cannot identify the effects — if any — of a conformal mapping on the convergence rate, which are related to coefficients varying between mesh points. If the coefficients' variation is too pronounced, other methods, which we will introduce in the context of boundary relaxation, have to be employed.

Systems of equations

We have up to now considered only scalar equations: the Laplace equation and the convection-diffusion equation form the building blocks of our formulation of the Navier-Stokes system.

As a final step, we consider the case of a system of equations. In this case, a full LFA analysis is still feasible but impractical, and is not covered here. We will instead use some of the concepts introduced in the previous sections to understand the principle underlying the relaxation of a linear system, as in the pressure form of the linearized Navier-Stokes system ($\mathcal{L}_\Delta$) given in section 3.1. We will arrive at the

conclusion that, in this particular case, a full LFA analysis is not strictly necessary and it is sufficient to consider the analysis already performed for the scalar cases to gain information on the behavior of a relaxation scheme applied to the full system.

We consider a linear system similar to the linearization of the two-dimensional Navier-Stokes equation in the form

$$L\,q = \begin{bmatrix} \mathcal{Q} + a_{11} & a_{12} & \partial_x \\ a_{21} & \mathcal{Q} + a_{22} & \partial_y \\ b_1 & b_2 & \Delta \end{bmatrix} q = f \tag{4.47}$$

where $\mathcal{Q} = \mathbf{U} \cdot \nabla - \nu\Delta$ is the linear convection-diffusion operator already presented in (4.30), and a_{ij} and b_j are scalar coefficients mimicking the base-flow velocity derivatives in the linearized Navier-Stokes equations. Similar to the scalar case, we define a splitting but, because we are dealing with a system, we have to consider two levels of splitting: (i) the splitting of the system and (ii) the splitting of the operators appearing as elements of the system. We represent the splitting of the system as

$$L_h^+ = \begin{bmatrix} \mathcal{Q}^+ + a_{11} & 0 & \partial_x^+ \\ 0 & \mathcal{Q}^+ + a_{22} & \partial_y^+ \\ 0 & 0 & \Delta^+ \end{bmatrix}_h , \tag{4.48a}$$

$$L_h^- = \begin{bmatrix} \mathcal{Q}^- & a_{12} & \partial_x^- \\ a_{21} & \mathcal{Q}^- & \partial_y^- \\ b_1 & b_2 & \Delta^- \end{bmatrix}_h \tag{4.48b}$$

where $\mathcal{Q}^+$, $\mathcal{Q}^-$ etc. represent the splitting of the discretized operators previously seen for the scalar analysis. Once the symbol of each scalar operator has been computed, we can analytically compute the inverse $\left(\tilde{L}_h^+\right)^{-1}$ and write the symbol $\tilde{M}_h$ of the iteration operator $M_h = -\left(L_h^+\right)^{-1} L_h^-$ as

$$\tilde{M}_h = \begin{bmatrix} \dfrac{1}{\tilde{Q}^+ + a_{11}}\left(\tilde{Q}^- - \dfrac{b_1\tilde{\partial}_x^+}{\tilde{\Delta}^+}\right) & \dfrac{1}{\tilde{Q}^+ + a_{11}}\left(a_{12} - \dfrac{b_2\tilde{\partial}_x^+}{\tilde{\Delta}^+}\right) & \dfrac{1}{\tilde{Q}^+ + a_{11}}\left(\tilde{\partial}_x^- - \dfrac{\tilde{\Delta}^-\tilde{\partial}_x^+}{\tilde{\Delta}^+}\right) \\ \dfrac{1}{\tilde{Q}^+ + a_{22}}\left(a_{21} - \dfrac{b_1\tilde{\partial}_y^+}{\tilde{\Delta}^+}\right) & \dfrac{1}{\tilde{Q}^+ + a_{22}}\left(\tilde{Q}^- - \dfrac{b_2\tilde{\partial}_y^+}{\tilde{\Delta}^+}\right) & \dfrac{1}{\tilde{Q}^+ + a_{22}}\left(\tilde{\partial}_y^- - \dfrac{\tilde{\Delta}^-\tilde{\partial}_y^+}{\tilde{\Delta}^+}\right) \\ \dfrac{b_1}{\tilde{\Delta}^+} & \dfrac{b_2}{\tilde{\Delta}^+} & \dfrac{\tilde{\Delta}^-}{\tilde{\Delta}^+} \end{bmatrix} . \tag{4.49}$$

In this expression, at first sight rather complicated, we have four types of terms: (i) the coefficients a_{ij} and h_j, which are independent of the mesh size h; (ii) the first-order derivatives ∂_x and ∂_y, whose symbol scales as $1/h$; (iii) the Laplacian operator, whose symbol scales like $1/h^2$ and (iv) the convection-diffusion operator, whose symbol scales as $1/h$ or $1/h^2$ depending on the Reynolds number Re_h.

As a consequence, if we consider again the limit of small mesh size h — equivalent to a small mesh-based Reynolds number Re_h — and recalling the results obtained for the convection-diffusion equations, namely that for low Re_h the dominant terms in the amplification factor are related to the diffusion term, the symbol $\tilde{M}_h$ simplifies to a diagonal matrix containing only the terms Q^-/Q^+ on the first two lines

and Δ^-/Δ^+ on the last line:

$$\lim_{h\to 0} \tilde{M}_h(\boldsymbol{\theta}) = \begin{bmatrix} \frac{Q^-}{Q^+} & 0 & 0 \\ 0 & \frac{Q^-}{Q^+} & 0 \\ 0 & 0 & \frac{\Delta^-}{\Delta^+} \end{bmatrix} \tag{4.50}$$

For a sufficiently small mesh size, the relaxation procedure defined by the splitting (4.48) behaves in the same way as the scalar convection-diffusion equation for the first two unknowns and as the scalar Poisson equation for the third. For even smaller Re_h, the entire system behaves like the Poisson equation since the amplification factor of the convection-diffusion equation tends to the one of the Poisson equation.

Boundary relaxation

As anticipated when the LFA analysis was introduced, there are two important cases where LFA analysis cannot be applied because the linearization of the operator is not meaningful: where the equation's coefficients vary too strongly between mesh points — for example, on coarse grids where boundary layers are poorly or not at all resolved — or in the vicinity of boundaries — where the discretization of boundary conditions takes the place of the interior equations. The results obtained so far cannot be applied to these regions: amplification factors closer to or even greater than one, denoting slow convergence or even divergence, are obtained even in the simple case of the Poisson equation with homogeneous Neumann boundary conditions if the pointwise Gauss-Seidel relaxation is used [14].

It has to be noted that while the slower error reduction in these region may affect the whole iterative process and require additional sweeps, its origin is essentially local, as elsewhere in the domain LFA analysis can be successfully applied. A possible solution is then to devise a special treatment, in the form of a different relaxation procedure like incomplete LU decomposition, GMRES or even direct algorithms, for the mesh points showing unacceptably slow error convergence. Because the number of degrees of freedom involved in these regions is commonly low — LFA analysis is usually valid in most of the domain — the extra computational cost is negligible when compared to the total cost. For the code used in this work, sparse LU decomposition [1, 2] is used to collectively solve all unknowns within a band of four mesh lines in the vicinity of the inflow and outflow boundaries and up to twenty mesh lines in the vicinity of the solid boundary. The larger number of mesh lines collectively solved close to the solid boundary has to be related to the strong variation of the equations' coefficients across the boundary layer developing there.

For a system of equations, boundary relaxation should include all unknowns (e.g. u, v, p for the two-dimensional Navier-Stokes equations) as the splitting of the system described in the previous section is not valid and the equations are different.

We will have a more detailed look at the effect of boundary relaxation when we consider some test cases in section 4.6.

4.4 Correction scheme
the linear equation

The analysis of the relaxation scheme we have performed so far has identified the amplification factor $\rho(M_h)$ associated with the discrete iteration operator M_h as the fundamental quantity used in analyzing the relaxation process. M_h is the result of a splitting $L_h = L_h^+ + L_h^-$ of the discrete problem L_h, and the correct design of this splitting is essential in order to obtain proper convergence rates for the error

amplitude. We have seen how the splitting corresponding to pointwise Gauss-Seidel relaxation represents a good choice for reducing the high-wavenumber error components for the Laplace equation and, provided that downstream marching is used, for the scalar convection-diffusion equation. We have also considered possible issues arising due to the presence of anisotropies in the problem, associated with both physical properties or grid stretching. Amplification factors for waves aligned with the largest mesh direction (wave vectors aligned with the finer) worsen with increasing anisotropy, but the original amplification factors can be recovered by employing linewise relaxation, which requires the implicit solution of tri- or pentadiagonal matrices. Finally, we have considered a particular system of equations similar in shape to the pressure form of the linearized Navier-Stokes equations ($\mathcal{L}_\Delta$) and shown that, for sufficiently small mesh size h — measured by the mesh-based Reynolds number Re_h — the smoothing properties for a given splitting of the system can be related to the scalar case. In addition, we remark that the low Re_h-limit corresponds to larger grids — and thus to larger numbers of degrees of freedom — and it is the more important limit as it relates to the case where most of the computational cost is invested.

We now return to the observation that both the pointwise and linewise Gauss-Seidel relaxation have very good amplification factors for high wavenumbers, but the amplification factor for low wavenumbers is close to one. The only exception is for high Reynolds number $Re_h \gg 1$ and downstream marching, but for any physical problem with a boundary layer that needs to be resolved, there will be at least one area of the domain where $Re_h = \mathcal{O}(1)$. An interpretation for the inferior convergence of the low-wavenumber error is linked to the idea that pointwise Gauss-Seidel iteration is a *local* process: at each iteration, information from one mesh point is passed only to the one next to it. As a consequence, transfer of information between two distant mesh points requires many iterations. While this is not an issue for purely convective (hyperbolic) problems, where there is a natural direction of information propagation — a property which is leveraged by downstream marching — it represents a bottleneck for elliptic problems, where each point in the domain influences all other points and information must be propagated back and forth until convergence. Linewise relaxation, despite the fact that it is global in the direction of mesh-lines that are solved implicitly, remains a local relaxation in the other direction, as can be seen in Figure 4.7, where contours are compressed in the vertical direction — corresponding to the direction of implicit solution — but do not change in the horizontal direction.

Many physical problems of interest contain at least an elliptic term — in our case it is the Laplace operator applied to the pressure in the pressure equation and to the velocities in the momentum equations. In this work, multigrid is the chosen algorithm to deal with the problem of low-wavenumber components. Alternative solutions would include more complicated relaxation methods like ILU decomposition [63, 36], which can be considered as a partially global iteration method, and Krylov subspaces methods, which change the basis on which the solution is searched in order to account for long-range interactions (small wavenumbers).

As we have earlier remarked, there are two possible multigrid algorithms: the Correction Scheme (CS) and the Full Approximation Scheme (FAS). Both are based on the idea of computing a correction to the approximation q_m of the exact, discrete solution $\bar{q}$ on coarser grids and then interpolate it to finer grids. They differ, however, in the way this correction is computed. In what follows, we start by introducing the more widely known CS scheme, after which we will present the FAS scheme.

The idea behind the Correction Scheme (CS) is to compute a correction e_h^m in order to obtain a better approximate solution $q_h^{m+1} = q_h^m + e_h^m$ at the next iteration of the iterative solver. By replacing the expression for q^{m+1} in a generic discrete linear system $L_h q_h^{m+1} = f_h$, it follows immediately that the

correction e_h^m satisfies the equation

$$L_h e_h^m = r_h^m, \quad \text{where} \quad r_h^m = f_h - L_h q_h^m. \tag{4.51}$$

In the above expression, r_h^m is the residual of the equation and is non-zero because q_h^m is merely an approximation to the exact discrete solution. The term e_h^m is an approximation of the error, given by $e_h = \bar{q}_h - q_h^m$.

If the high-wavenumber components of the error have been reduced by the application of some (usually one or two) sweeps of a variant of the Gauss-Seidel relaxation process (described in section 4.3), the correction e_h^m can be approximated on a coarser grid by the function e_H^m satisfying

$$L_H e_H = R r_h \tag{4.52}$$

where R is the restriction operator used to transfer information from grid h to grid H, and the superscript ^{m} has been omitted. The computational cost of solving this equation for the correction on the coarse grid is clearly less than the cost of solving it on the finer. Once a solution e_H to the coarse-grid equation (4.52) is found, it can be interpolated back to the finer grid, and the approximate solution $\mathbf{q}_h^m$ is updated as follows

$$q_h^{m+1} = q_h^m + I e_H^m. \tag{4.53}$$

The interpolation process introduces some high-wavenumber error components which can be efficiently reduced by one or more additional relaxation sweeps. This two-grid scheme can be implemented recursively by repeating the same procedure on increasingly coarser grids, until a grid is reached on which the direct solution of (4.51) is inexpensive.

The Correction Scheme can be visually summarized with the help of Figure 4.8, where four grid-levels have been considered, as in Figure 4.1: the simplest multigrid algorithm starts from the finest grid (orange dots), where one or two relaxation sweeps are applied to reduce the high-wavenumber error components. The residual of the discrete equation is then restricted onto the coarser grid (green dots) where the equation for the correction e_H^m (4.52) is defined and relaxed. This procedure is repeated recursively until the coarsest grid (red dots) is reached, where an exact discrete solution of the corresponding equation for the correction can be obtained inexpensively. After that, corrections are interpolated back to finer grids using equation (4.53). At every grid level in the upward, i.e., coarse-to-fine, leg the equation for the correction is relaxed again with one sweep of the relaxation process to reduce the high-wavenumber components introduced during the interpolation process. On the finer level, the solution is updated. While it is possible to perform LFA analysis for the two-grid (and consequently for the multi-grid) problem, it is not attempted here. Suffice it to remark that, on each grid, it is necessary to reduce only the high-wavenumber error components since lower-wavenumber components will be reduced on coarser grids. As a consequence, we can expect the limiting amplification factor of each application of the multigrid V-cycle — i.e. one downward and one upward leg in Figure 4.8 — to be dependent on the largest amplification factor $\rho(\boldsymbol{\theta})$ in the high-wavenumber range determined in section 4.3. For all cases analyzed in section 4.3 the maximum amplification factor for the Gauss-Seidel iteration in the high-wavenumber range is approximately 0.5, provided that downstream marching and linewise relaxation is used if necessary (note that it is exactly 0.5 for pointwise Gauss-Seidel relaxation on an isotropic grid). If, like in the case we just considered, we apply a total of three relaxation sweeps per grid level — two on the downward leg and one on the upward leg — the upper limit for the amplification factor for each V-cycle is $0.5^3 = 0.125$, meaning that the magnitude of the error is expected to be reduced by nearly one

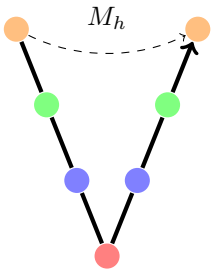


Figure 4.8: The V-cycle is a main ingredient of the multigrid algorithm: on the downward leg, some — usually one or two — relaxation sweeps are applied at each grid level (represented by the colored circles) except the coarsest (red circle), and information is transferred from the finer to the coarser grids by restriction. The equation defined on the coarsest grid is solved exactly. Corrections are then interpolated back to finer grids on the upward leg, and additional relaxation sweeps — usually only one — are applied to reduce the high-wavenumber error introduced by the interpolation process. The whole process can be seen as the application of an iteration operator M_h, and a spectral radius $\bar{\rho}(M_h)$ can be computed.

order of magnitude for each V-cycle iteration.

The value of 0.125 is the same we have obtained during the analysis of the Lexicographic Gauss-Seidel relaxation for the Poisson equation. The important difference is that, while in the former case it was the worst amplification factor for high-wavenumber components — far worse amplification factors have been obtained for lower wavenumbers — in the present multigrid case it is the worst amplification factor for *all* wavenumbers.

We can then return to the global point of view and summarize all sequential operations in the V-cycle, represented in Figure 4.8, by a single operator M_h acting on the error components in the same manner as has been described when defining a generic iterative process at the beginning of this chapter (see equation (4.4) in section 4.1). In the same way we can identify the asymptotic behavior of the iterative process represented by the operator M_h with its spectral radius $\bar{\rho}$, which is also the worst obtainable convergence rate: for the multigrid process represented in Figure 4.8 the spectral radius is thus equal to 0.125.

Full multigrid (FMG) and Full V-cycle (FV)

The V-cycle represented in Figure 4.8 starts from an initial guess on the finest grid. A smart way of computing this initial guess is by interpolating solutions previously obtained on coarser grids. This procedure leads to the introduction of the Full Multigrid Algorithm (FMG), which differs from the iterative algorithm presented above in the fact that the initial guess is inexpensively computed on the coarsest grid instead of the finest grid. Once a solution is obtained on this grid, an initial guess for the next finer grid can be obtained by interpolation, and a multigrid cycle is applied until a converged solution is found on this finer grid. This process is then repeated with progressively finer grids until a satisfactory mesh size is obtained.

The availability of a good approximation with which to start the V-cycle reduces the number of full V-cycles required to converge to the required solution and, for simple problems like the Poisson equation — or even more complex ones like the inviscid, incompressible Navier-Stokes equations — it has been shown that one V-cycle is sufficient to obtain a solution whose algebraic error e_h falls below the discretization error [14].

Finally, a combination of the V-cycle with the FMG algorithm results in the definition of the FV-cycle where, given an initial approximation on the finer grid — for example obtained when an additional grid

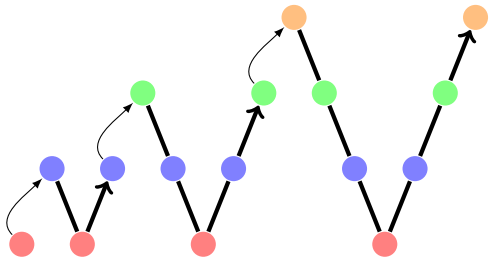


Figure 4.9: In the FMG algorithm, the initial guess used at the finest level of a V-cycle is computed by interpolating a solution previously computed on a coarser grid. Progressively finer grids can be added until a satisfactory mesh size is obtained. The availability of a good initial guess computed by interpolation reduces the number of V-cycles required to converge to the solution: in many cases one V-cycle is sufficient to converge to algebraic error below the discretization error. The full solution has to be interpolated when a new grid level is added (thin, curved arrows), while we recall that only the corrections are interpolated inside each V-cycle (thick, straight arrows).

is added in an FMG algorithm — a downward leg is applied to reach the coarsest grid used. After that, an FMG algorithm can be used to get back to the finer grid, thus reducing the number of relaxation sweeps required on the finest grid. The FV cycle is represented in Figure 4.10.

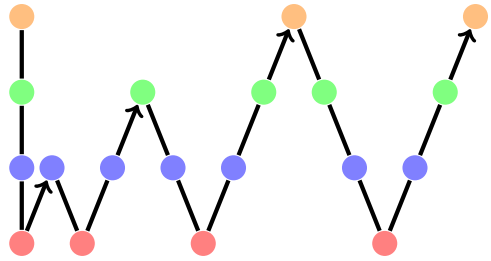


Figure 4.10: The FV-cycle can replace the V-cycle as a standalone algorithm or inside an FMG algorithm. The initial guess on the finest grid (orange) is relaxed and restricted until the coarsest grid (red) is reached, in the same way as in the downward leg of a V-cycle. A path similar to the FMG algorithm is then used, with the difference that an approximation of the solution is already available on all grids and only the corrections are interpolated in the upward legs. The FV-cycle is intended to reduce the number of sweeps applied to the finest grid.

Both FMG and FV cycles are used in the code developed in the context of this work, but the Correction Scheme is replaced with the Full Approximation Scheme to be able to accommodate equations — or a system of equations — containing nonlinear terms and to include adaptive refinement procedures.

In the next section the FAS algorithm will be introduced. After that, the last section of this chapter will provide some test cases with applications and demonstrations of the theoretical results obtained in the previous sections.

4.5 Full Approximation scheme
the non-linear equation

The Full Approximation Scheme may be less known but is certainly the most powerful version of multigrid. Its main advantages are the capability of directly addressing nonlinear problems and of providing a natural approach to adaptive grid refinement.

The Full Approximation Scheme replaces the coarse grid equation (4.51) for the correction e_h with a

coarse grid equation for the unknown q_H in the form

$$L_H q_H = f_H + \tau_h^H \tag{4.54}$$

where $f_H = R f_h$ is the restriction of the forcing from the fine to the coarse grid, and τ_h^H is the defect correction defined as

$$\tau_h^H = L_H \left(R q_h\right) - R \left(L_h q_h\right). \tag{4.55}$$

Once a solution q_H is computed on the coarse grid, only the correction (and not the whole solution) is interpolated back to update the fine grid solution according to

$$q_h^{new} = q_h + I \left(q_H - R q_h\right). \tag{4.56}$$

Interpolation of the full solution by writing $q_h^{new} = I q_H$ would introduce the interpolation error of the whole solution instead of the interpolation error of only the correction and is thus not advised (but see [8, section 8.5] for cases when this is not true).

To better understand the meaning of the defect correction, we start by considering the error introduced by the discretization process, i.e. by moving from the continuous to a discrete formulation. This error is in general unknown but is proportional to the order of the discretization. For example, a centered second-order derivative can be written on a one-dimensional domain as

$$q_x = f \quad \Longrightarrow \quad \frac{q_{i+1} - q_{i-1}}{2h} + f\left(x\right) h^2 = f_i. \tag{4.57}$$

Let us suppose for a moment that the function $f\left(x\right)$, representing the error introduced in the discretization process and independent of the mesh size, is known: the discretized equation could then be written as

$$\frac{q_{i+1} - q_{i-1}}{2h} = f_i - f\left(x\right) h^2, \tag{4.58}$$

and the discrete solution would correspond to the analytic one independently of the chosen mesh size.

While we cannot know *a-priori* the shape of the function $f\left(x\right)$, the equivalent information is readily available when considering two different discrete representations on two grids of different mesh size h and H. To see this, we rewrite τ_h^H (4.55) as

$$\tau_h^H = \underbrace{L_H \left(R q_h\right) - f_H}_{r_H} - \underbrace{R \left(L_h q_h - f_h\right)}_{R r_h} \tag{4.59}$$

where $f_H = R f_h$ by definition. The first term in this equation represents the residual stemming from the application of the coarse grid operator L_H on the (restricted) fine grid solution q_h, while the second term represents the restriction of the fine grid residual, thus providing the difference in the application of the coarse grid operator L_H and the fine grid operator L_h on the same fine grid solution q_h. If this difference is added to the coarse grid equation, the same solution is obtained on both grids.

We also mention the possibility, not used in the code developed for this work, of extrapolating the value of τ_h^H to compute a solution as close as possible to the solution of the continuous problem instead of the one to the discrete problem. In other words, the defect correction τ_h^H can be used to estimate the unknown function $f\left(x\right)$ — or its multi-dimensional equivalent — defining the truncation error of the discretization process. This can be used to force the finest grid equation, for which the defect correction $\tau_\cdot^h$ is not available. A concise description of this methodology can be found in [8, section 8.4].

Nonlinear equations

The correction equation $L_H e_H = r_H$ (4.52) used in the definition of the Correction Scheme algorithm is valid only for linear problems and, as a consequence, limits the applicability of the CS to linear problems. This same limitation does not apply to the FAS algorithm, as the correction is computed for the equations by means of the defect correction τ_h^H and not for the solution. This is equivalent to writing the correction equation in the form

$$L_h\left(u_h^m + e_h^m\right) - L_h\left(u_h^m\right) = r_h$$

and replacing the fine grid operators L_h with their coarse grid counterparts L_H. After u_h^m, e_m^h and r_h are consistently multiplied by the restriction operator R, an equation equivalent to the FAS equation (4.54) is obtained. Finally, it should be noted that the Correction Scheme can be used as the linear solver within a Newton algorithm, but the FAS algorithm is more efficient as it does not require an external Newton iteration on the finest grid.

Adaptive grid refinement in the context of multigrid

A possibly even more interesting feature of the FAS algorithm is its ability to naturally treat adaptive grid refinement. Because the solution, in contrast to the correction, is represented on all grids, the finer grid is not required to have the same extent of its next coarser grid, as shown on the finest grid of Figure 4.1. In this case, the defect correction τ_h^H will be defined on the coarse grid only in the areas where a finer grid is defined and will be zero otherwise but will nonetheless affect the solution on the entire coarse grid. Because τ_h^H is an estimate of the truncation error, this is a natural approach: when a region of a given grid does not need to be refined, it means that its truncation error is sufficiently small and can be taken as zero.

A question arises on how to deal with inner boundaries, i.e., those boundaries of the finer grid that do not correspond to the physical boundaries of the domain but lie in the interior of the coarser grid. A simple and correct answer is to use Dirichlet conditions obtained by interpolating the coarse grid solution with an interpolation operator of at least the order of the discretization. The reason for this approach follows the same argument given above: when a region of a given grid does not need to be refined, it means that the chosen discretization order represents the solution sufficiently well in that region. An interpolation of the same order as the discretization will then satisfy the fine grid equations as well. For example, when a second-order discretization is employed, we can stop further refining when, locally, the solution is well approximated by a second-order polynomial (a parabola in one dimension or a paraboloid in multiple dimensions). In this sense, a second-order interpolation of the solution would locally be an exact solution of a finer grid and can be used as a Dirichlet condition for the refined region.

An additional remark has to be made on the use of the defect correction τ_h^H as a refinement criterion, taking advantage of the fact that it is an approximation of the truncation error. Under this criterion, refinement is then required only where τ_h^H is large, corresponding to the area where a stronger modification of the coarse grid equations is induced by the presence of the finer grid. The defect correction refinement criterion is both less expensive — as it is a byproduct of the FAS algorithm — and more appropriate than many, more commonly used refinement criteria based on solution gradients or the vorticity field — which requires the additional computation of the gradients and fails to identify regions of interest in very simple cases. As an example where gradient-based adaptive grid refinement criteria would fail, we can consider a problem with a solution given by a parabolic profile in the form

$$q = ay^2 + by + c,$$

and a second-order discretization of the problem. An example could be plane Poiseuille flow between two stationary flat walls driven by an externally imposed pressure gradient, for which the velocity profile is

$$u = -\frac{y}{\nu} p_x \left(h - \frac{y}{2} \right) \tag{4.60}$$

with h as the channel half-height. With the velocity gradient stronger close to the wall, a gradient-based refinement criteria would suggest to concentrate mesh points in this region. In reality, this would be ill-advised in terms of computational resources, as three mesh points is all that is needed to obtain an exact solution to the problem: the second-order scheme describes parabolic solutions exactly. To see this we define a second-order discretization of the u-momentum equation on a three-point mesh of size $h = b$ as

$$\nu \frac{u_{i+1} - 2u_i + u_{i-1}}{h^2} = p_x \tag{4.61}$$

where the pressure gradient p_x is given. No-slip boundary conditions at the walls impose $u_{i-1} = u_{i+1} = 0$ and the equation reduces to

$$-2\nu \frac{u_i}{h^2} = p_x \qquad \Longrightarrow \qquad u_i = -\frac{h^2 p_x}{\nu}, \tag{4.62}$$

which is exactly the velocity at the centerline obtained from (4.60) for $y = h$. The velocity at all other locations can be computed exactly by second-order interpolation of u_i at $y = h$ and $u_{i-1} = u_{i+1} = 0$ at $y = 0$ and $y = 2h$.

The most consistent criteria to be used is then to refine where the local solution to the problem is farthest away from the shape implied by the discretization scheme — a parabola in the case of a second-order scheme. In this sense, the defect correction gives an indication of how well the solution is locally described by the discretization scheme.

4.6 Test cases

We will now investigate some applications of the computational tools presented so far. The multigrid code used to produce the following examples is written in Matlab and listed in Appendix A. Both the CS and the FAS scheme have been implemented, and results from the two schemes are the same up to numerical precision. The equations are discretized using a standard five-point second-order Laplacian and a first-order upwinded convective term. Pointwise Gauss-Seidel and linewise Gauss-Seidel iterations are implemented by operator splitting to keep a compact notation. Full-weighted restriction and bilinear interpolation are used for the communication between grids. For all test cases the finest grid has 129×129 mesh points, the coarsest one has 5×5 mesh points.

In these tests, particular attention is directed towards the boundary treatment: this part of the multigrid process cannot be analyzed by LFA and, as noted by Diskin [14], has been often overlooked. Following Diskin's steps, we will see how boundary relaxation restores the convergence rates predicted by LFA analysis and extend his analysis of the inviscid equations to the case of the convection-diffusion equation. This section contains four figures, representing the convergence history of both pure Gauss-Seidel iterations (blue lines) and of Gauss-Seidel iteration as a relaxation method within multigrid (red lines). Each line symbol corresponds to three iterations of the Gauss-Seidel algorithm (blue) or one V-cycle of the multigrid algorithm (red), since three GS iterations are performed for each V-cycle on the finest grid. Figure 4.11 shows results for the Poisson equation with Dirichlet boundary conditions on

an isotropic grid. Figure 4.12 shows results for the Poisson equation with Dirichlet boundary conditions on an anisotropic grid and the effect of linewise Gauss-Seidel relaxation. Figure 4.13 shows the effect of changing the boundary condition from Dirichlet to Neumann and how boundary relaxation can be used for recovering a satisfactory convergence rate. Finally, Figure 4.14 shows results for the convection-diffusion equation. The obtained results match very well the theoretical estimates based on LFA analysis, despite the fact that LFA relies on the hypothesis of an infinite grid and constant coefficients.

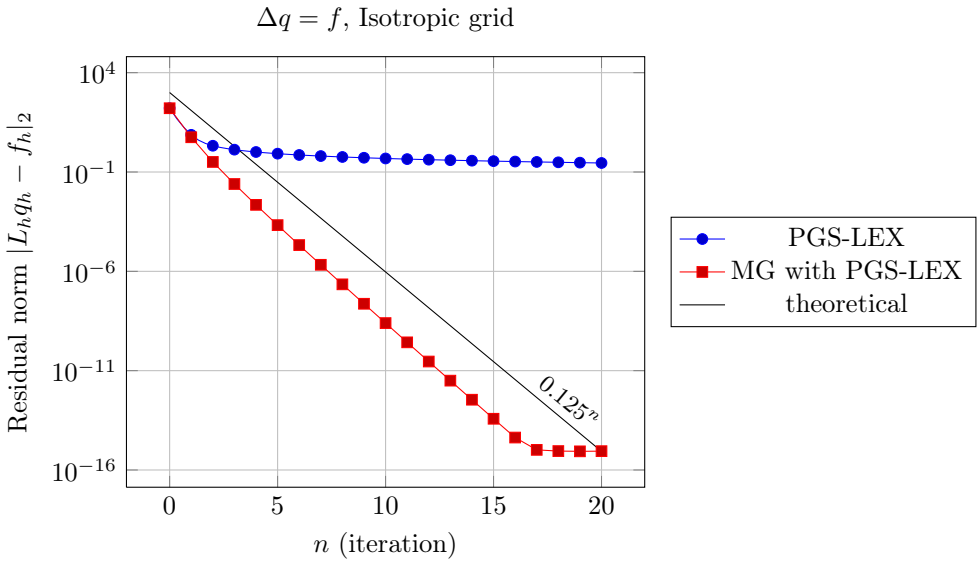


Figure 4.11: Convergence history of the residual for Lexicographic Gauss-Seidel relaxation (GS-LEX, blue circles) and multigrid cycles (MG, red squares), using semilogarithmic axes. Only every third GS-LEX iterations is marked by a symbol along the blue line, and three GS-LEX iteration per grid are applied for each V-cycle; this makes the two curves nearly comparable in terms of computational cost. The Poisson equation with Dirichlet boundary conditions is discretized on a uniformly spaced grid. The finest grid, on which the residual is computed, has 129×129 mesh points, the coarsest grid has 5×5 points. Three sweeps of GS-LEX relaxation are used on each grid level, two during the downward leg and one during the upward leg. The theoretical asymptotic convergence (spectral radius) of GS-LEX and MG are $\bar{\rho}_{GS} = 1$ and $\bar{\rho}_{MG} = 0.5^3 = 0.125$, respectively, and the theoretical convergence history for MG is indicated by a black line (it is a horizontal line for GS-LEX). While the decrease in residual norm associated with the first three GS-LEX and the first V-cycle iterations are comparable (at $n = 1$ the curves coincide), the far superior convergence rate of MG is already evident starting with the second iteration $n = 2$. Excellent agreement with the results from LFA analysis is obtained.

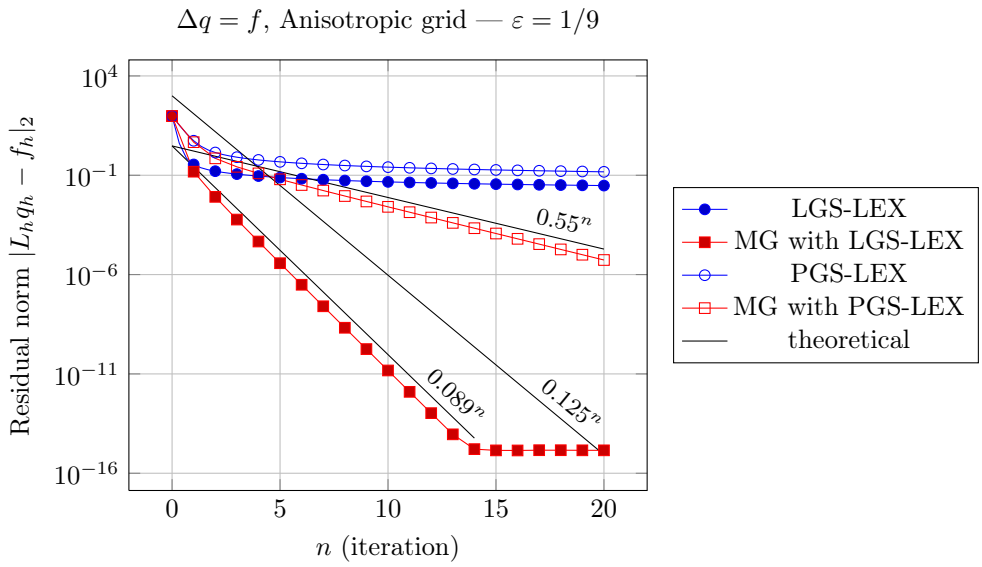
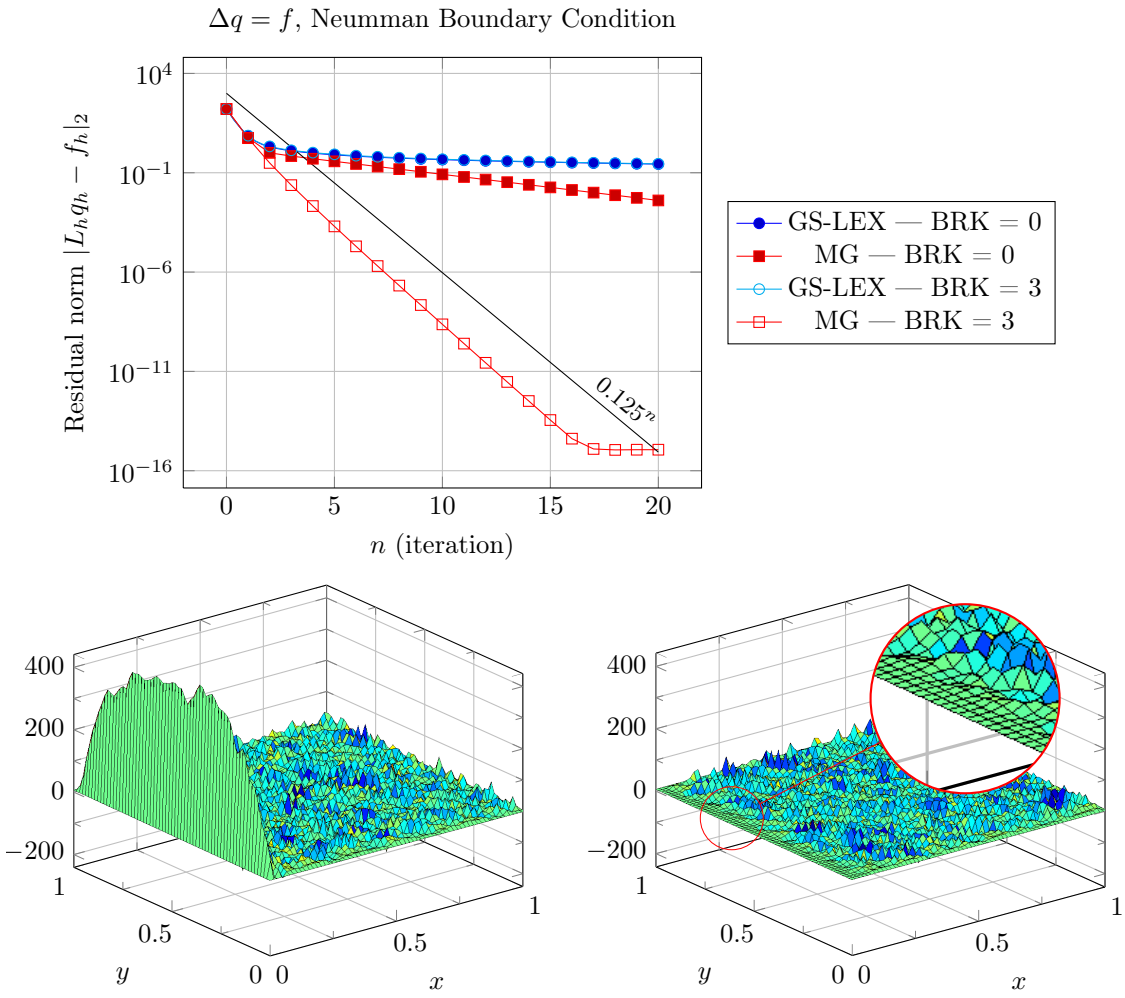


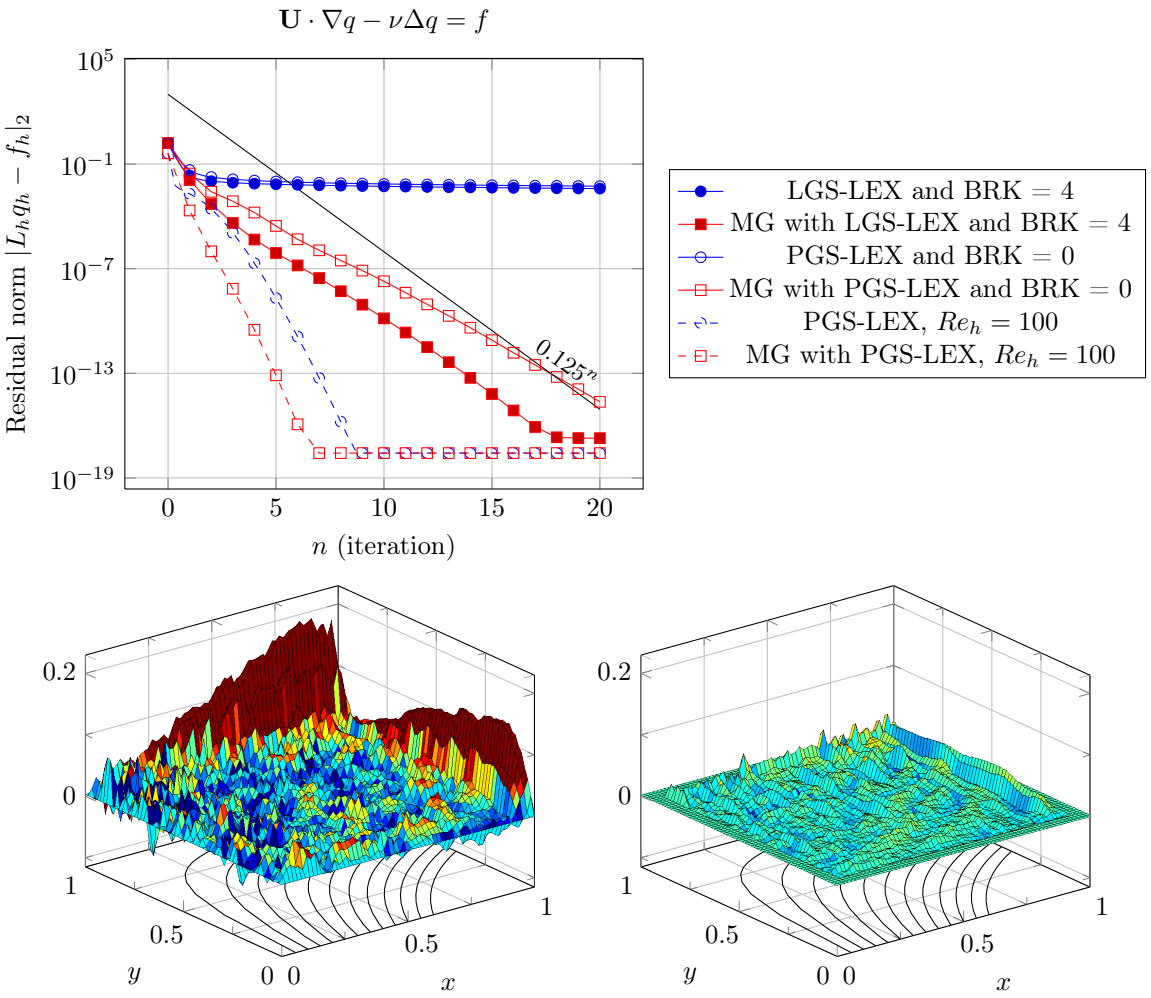
Figure 4.12: Same as Figure 4.11, but with an anisotropy of $\varepsilon = 1/9$, corresponding to a grid stretching of $h_2/h_1 = 1/3$. Results for pointwise Gauss-Seidel (PGS-LEX) are shown with empty symbols, results for linewise Gauss-Seidel (LGS-LEX) are indicated with filled symbols. This small stretching ratio is sufficient to cause a severe degradation in the multigrid asymptotic convergence rate of PGS-LEX: LFA analysis estimates a V-cycle convergence rate of 0.55, and our results (red, empty squares) show excellent agreement with this estimate. The use of linewise Gauss-Seidel as a relaxation method recovers — and improves — the convergence rate obtained for the isotropic grid: the theoretical estimate for LGS-LEX, at 0.089 per V-cycle, is sligthly better than the 0.125 of the PGS-LEX on an isotropic grid and, as in the previous figure, there is excellent agreement between the theoretical estimate and the numerical experiments.



(a) without boundary relaxation

(b) with boundary relaxation

Figure 4.13: Same as Figure 4.11, but with a homogeneous Neumann boundary condition on the $x = 0$ boundary. Pointwise Gauss-Seidel relaxation is used for all curves. The replacement of the Dirichlet with a Neumann boundary condition gives rise to a severe degradation in the convergence rate of the multigrid V-cycle, whose converged history is showed in red, unfilled squares. The addition of a boundary relaxation solving all points belonging to the first three mesh lines close to the $x = 0$ boundary recovers the convergence rate of 0.125 per V-cycle predicted by LFA analysis. The effect of boundary relaxation is clearly visible in the (a) and (b) subfigures, showing the residual of the equations after the application of two V-cycles without and with boundary relaxation, respectively. PGS-LEX performs poorly in reducing the error on the boundary, and a spike in the error is clearly visible in the left subfigure (a). The positive effect of boundary relaxation is evident in the right subfigure (b), where the spike has disappeared and the residual on the first three mesh lines close to the boundary is at machine precision: the direct (or iterative) solution of this region adds very little to the overall cost of the solution procedure but allows us to restore the theoretical multigrid convergence rate. It can also be noted that pure GS-LEX iteration are barely modified by the addition of boundary relaxation: the filled blue and empty cyan circles are hardly distinguishable. This suggests that boundary relaxation acts at the restriction and interpolation levels rather than at the relaxation level, by removing the high-wavenumber components close to the boundary, left behind by PGS-LEX.



(a) without boundary relaxation

(b) with boundary relaxation

Figure 4.14: Convection-diffusion equation for $Re_h = 1$. Streamlines of the velocity field are plotted in the bottom figures: inflow and outflow are at the bottom right $y = 0$ and at the top right $x = 1$ boundary, respectively, a stagnation point is located at $(x, y) = (0, 1)$, and the flow is symmetric with respect to the $x = 0$ axis. Lexicographic relaxation corresponds to marching in the downstream direction. Convergence of the multigrid algorithm with pointwise Gauss-Seidel relaxation and no boundary relaxation is shown in red, unfilled squares. Filled squares correspond to multigrid with linewise Gauss-Seidel and boundary relaxation applied to four mesh lines from all boundaries. Subfigures (a) and (b) show the residual after the application of two V-cycles for PGS-LEX without boundary relaxation and LGS-LEX with boundary relaxation; the situation is similar to the case with a Neumann boundary condition, but the origin is different: larger residuals are found close to the solid boundary and at the outflow where, as on the rest of the boundary, a Dirichlet condition is applied. The origin of this larger residual can be found in the chosen discretization: the upwinded convective term, propagating information only from upstream, conflicts with the centered diffusive term, propagating information also from the boundary. Because the lexicographically ordered sweeps naturally transfer information downstream, the residual is "accumulated" close to the solid boundary and the outflow. In the case of the outflow, a better implementation of the boundary condition can partly fix the problem but, in both cases, boundary relaxation can be used to propagate the information from the boundary into the domain. Cases with both higher and lower Re_h do not show this problem, suggesting that it is associated with intermediate Reynolds numbers only. As an example, the convergence histories of lexicographic pointwise Gauss-Seidel and of the corresponding multigrid V-cycle are shown in blue and red dashed lines, respectively.

Chapter 5

Global Analysis of the Flow Around a Leading Edge

In this chapter we will present results from the global analysis of the flow around a leading edge, based on the theory outlined in chapter 2 and the numerical approach described in chapter 3 and chapter 4.

A brief recall of the governing equations and of the numerical algorithms will lead us to the description of the main features of the base flow.

Our global analysis is performed at a chord-based Reynolds number $Re_C = 10^6$, corresponding to a radius-based Reynolds number $Re_r = r\,Re_C = 16000$ and a sweep Reynolds number $Re_s = \sqrt{Re_r}\tan\Lambda = 126$. The dimensionless leading-edge radius r and sweep angle Λ for our geometry are $r = r^*/C^* = 0.016$ and $\Lambda = 45°$ ($\tan\Lambda = 1$), respectively. Small perturbations to a spanwise-independent solution to the Navier-Stokes equations ($\mathcal{R}$) for this set of parameters are known to decay asymptotically in time: previous stability analyses on simplified geometries [38, 39, 48] or for the supersonic case [43] consistently suggest a critical sweep Reynolds number Re_s of about 600 for attachment-line instabilities to develop. Maintaining the current configuration, a sweep Reynolds number Re_s of 600 would correspond to a chord-based Reynolds of $Re_C = Re_s^2/r = 22.5 \cdot 10^6$, which is beyond our numerical capabilities for the time being. We recall the definition of the various Reynolds numbers from section 2.1:

$$Re_C = \frac{U_\infty^* C^*}{\nu^*}, \qquad Re_r = \frac{U_\infty^* r^*}{\nu^*}, \qquad Re_s = \frac{W_\infty^* \delta^*}{\nu^*}.$$

Consequently, a stable spectrum is computed under the assumption that its main features will not change qualitatively when crossing the critical Reynolds number. Comparison between our results and the literature just cited corroborates this line of thought. In particular, the shape of the computed eigenvectors recovers features already observed by Mack et al. [44], namely a connection between the attachment-line instability and the crossflow instability. The modal structures at the attachment line closely resembles the findings of Lin & Malik [38].

As outlined in chapter 2, we are interested in the receptivity of the spectrum to a forcing of the perturbation equations, and this receptivity is related to the adjoint field. The adjoint spectrum is computed and shown to be equivalent to the direct spectrum (up to complex conjugation), provided some care is taken in dealing with the boundary conditions to avoid numerical difficulties. Analysis of the adjoint modes shows major receptivity of the corresponding direct modes to be concentrated in the upstream part of the domain, in particular, in the region close to the attachment line.

In the last section of this chapter, the wavemaker is described for our configuration, together with some observations on the structural sensitivity including its consequence on the effective implementation of active and passive control strategies.

5.1 Base flow

The baseflow is computed as a steady-state solution of the unforced Navier-Stokes equations ($\mathcal{L}$) around the leading edge of a Joukowsky airfoil. The Navier-Stokes equations read

$$\mathcal{R}(\mathbf{q}) \equiv \begin{cases} \partial_t \mathbf{u} + \nabla \mathbf{u}\,\mathbf{u} - \nu \Delta \mathbf{u} + \nabla p &= 0 \\ \nabla \cdot \mathbf{u} &= 0 \end{cases} \qquad (\mathcal{R})$$

where the dimensionless kinematic viscosity ν is the inverse of the Reynolds number. Boundary conditions are specified as the no-slip condition $\mathbf{u} = 0$ on the solid boundary, the inviscid solution $\mathbf{u} = \mathbf{U}_{\nu=0}(x, y)$ at the inflow and a homogeneous Neumann condition $\partial \mathbf{u}/\partial n = 0$ at the outflow. The shape of the airfoil is defined by a complex mapping from a rectangular domain, and the inviscid solution $\mathbf{U}_{\nu=0}$ can be obtained accordingly. The configuration is shown in Figure 2.1. See chapter 3 for all details on the mapping and the computation of the inviscid solution.

For numerical convenience, the mass conservation equation $\nabla \cdot \mathbf{u} = 0$ is replaced by a Poisson equation for the pressure [21, 56] obtained by the application of the projection operator to the Navier-Stokes system ($\mathcal{R}$) [65, 64, 14]. The application of the projection operator is equivalent to the computation of the divergence of the momentum equations and the application of the divergence-free condition. A full account of the projection and its consequences have been provided in chapter 3. We recall here that equivalence between the original and the projected formulation is guaranteed only if the divergence-free condition is used to obtain a non-homogeneous Neumann boundary condition for the pressure [64]. This can be accomplished by manipulating the momentum equations in the direction normal to the boundary in order to construct a Neumann boundary condition for the pressure: the divergence-free condition on the boundary is then implemented by using a finite-volume-like formulation for the diffusive term, see chapter 3 for details. Because the two formulations are equivalent and the Poisson equation is only a numerical expedient, we will always refer to our solutions as the solutions of the system in divergence form ($\mathcal{R}$).

A spanwise invariant solution of ($\mathcal{R}$) is sought by removing all the spanwise z-derivatives. Under the hypothesis of spanwise invariance, the solution is known to be stable at all Reynolds numbers, and a steady-state solution can easily be obtained. A consequence of the spanwise invariance is that u, v and p are decoupled from the spanwise velocity component w and can be solved independently. The w-component of velocity can be computed at a later stage and is governed by a scalar, linear convection-diffusion equation.

Computation of the base flow

The governing equations ($\mathcal{R}$) are discretized on a stretched grid covering approximately 20% of the chordwise extent of the profile with a second-order finite-difference scheme. Upwinded stencils are used for the convective terms $U\partial_x + V\partial_y$ — the term $W\partial_z$ is zero because of the spanwise invariance — and centered stencils for all other derivatives. The Laplacian operator is discretized using a finite-volume formulation so that $\Delta u = A^{-1}\Sigma_i \nabla u \cdot \mathbf{n} ds$, where the sum is over the boundary of the control cell Ω and A is the surface of Ω. The gradients of the velocity components are evaluated on the boundary of the control cell. Details of the discretization and its validation are given in chapter 3.

The discretized, unforced equations are solved using the iterative, multigrid-based DNS code introduced in chapter 4. In order for the multigrid solver — or any Newton-type solver — to converge to a

correct solution, a good initial guess has to be provided. Consequently, solutions to ($\mathcal{R}$) are obtained by continuation over a set of Reynolds numbers ranging from $Re_C = 10^3$ to $Re_C = 10^6$: all computations except the first ($Re_C = 10^3$) take the solution at the previous Reynolds number as an initial guess. A third order interpolation is used when an additional, finer grid is added to the grids stack. The first computation takes the inviscid solution as an initial guess. The increase in the Reynolds number is performed manually to verify the convergence of the solver at each value of the Reynolds number.

A grid stretching is applied in the direction normal to the solid boundary to cluster more mesh points into the boundary layer area. The number of mesh points on the finest grid is increased with the Reynolds number to maintain a nearly constant number of points across the boundary layer thickness. The solution at the highest Reynolds number of $Re_C = 10^6$ ($Re_r = 16000$, $Re_s = 126$) contains nearly 40 points across the δ_{99}-thickness at the attachment line.

In the following two section, the multigrid algorithm is characterized by specifying the choice of a relaxation procedure and of a multigrid scheme.

Relaxation procedure

A linewise Gauss-Seidel relaxation procedure is employed: the linearized discretized equations belonging to a given variable — say, the pressure p — and to a given line along the coordinate direction parallel to the solid boundary are solved together, starting from the inflow boundary and marching downstream towards the solid boundary. One or more sweeps are applied before moving to the other variables. The same procedure is then applied for u, v, w.

For each line, the solution of a penta-diagonal linear system is required, and LAPACK [3] routines for banded matrices are employed. The penta-diagonal system corresponds to the one-dimensional discretization of the Laplacian operator in the case of the pressure equation and to the upwinded discretization of the convection-diffusion operator $\mathcal{Q}_\nu$ (2.9) in the case of the momentum equations. A collective solution of all variables u, v, w, p is performed by employing an LU decomposition for mesh points within a range of four mesh lines from the inflow and outflow boundaries and within 20 mesh lines from the solid boundary to avoid difficulties in the convergence of the relaxation process due to the different discretization stencils used at the boundaries and strongly varying coefficients in the boundary layer, in particular, on coarser grids.

Multigrid scheme

A multigrid Full Approximation Scheme (FAS) multigrid scheme with adaptive grid refinement is employed. FAS differs from the more widely known Correction Scheme (CS) in storing the full solution on all grid levels instead of the corrections to the solution on the finest grid. As has been shown in chapter 4, the FAS algorithm has two main advantages over the CS: the capability of dealing with nonlinear equations without the necessity of an outer Newton-like iteration and the possibility of handling adaptively refined grids.

An initial guess for the solution is provided on each grid which is available from previous computations — for the first computation, an initial guess is provided on the coarsest grid. Third-order interpolation is used to provide an initial guess when a finer grid is introduced. A series of V-cycles is then applied, with two sweeps of the relaxation procedure described in the previous section applied on each grid on the downward leg and one sweep per grid on the upward leg. At the coarsest level, the discretized problem is solved to numerical precision by means of a Newton method based on an LU decomposition of the Jacobian matrix.

Transfer of information between grids is performed using a full-weighting operator when moving from a finer to a coarser grid (downward leg) and a third-order interpolation when interpolating corrections from a coarser to a finer grid (upward leg).

The convergence behavior of the multigrid solver for a selection of Reynolds numbers and grid sizes is shown in Figure 5.1 and compared with the theoretical convergence rate (black line) for the two-dimensional, scalar, constant-coefficient Laplace equation on a uniformly spaced grid using a pointwise lexicographic Gauss-Seidel relaxation as described in section 4.3. While the computational cost for the two cases is not directly comparable due to the fact that the line Gauss-Seidel relaxation is somewhat more expensive — the LU decomposition of the penta-diagonal system is not necessary in the point-wise Gauss-Seidel relaxation — the comparison gives a good indication on the total number of iterations that are required to reach a numerically converged solution. The actual convergence closely matches the convergence for the scalar Laplace equation for most of the cases. The degradation in the convergence rate when increasing the Reynolds number can be associated with the fact that the coarsest grid used in the computation is no longer able to resolve the boundary layer, whose thickness becomes smaller than the mesh size. When moving from $Re_C = 2 \cdot 10^5$ to $Re_C = 5 \cdot 10^5$ it was necessary to increase the number of points on the coarsest grid from 129×33 to 257×65, as the multigrid algorithm would not converge otherwise.

The last computation shown on the right of Figure 5.1, which provides the base flow used for the stability analysis that will be presented later in this chapter, is an example of adaptive refinement. The finest grid of size 8193×1025 covers only the half closest to the solid boundary of the next coarser grid, as shown in Figure 3.7. The principles behind multigrid adaptive refinement have been outlined in section 4.5.

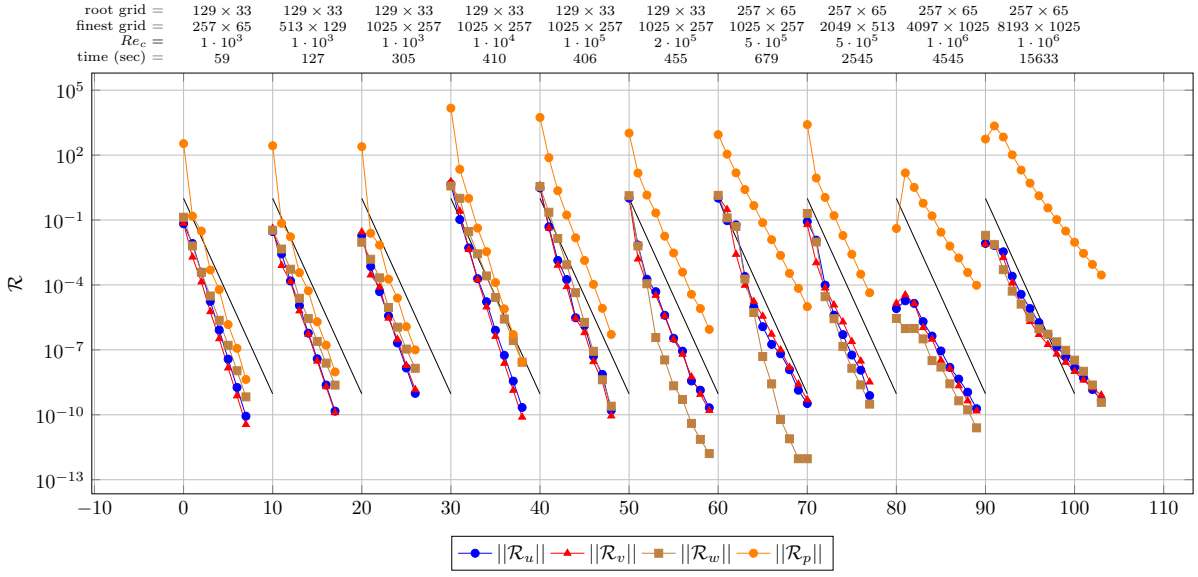


Figure 5.1: Convergence behavior for the computation of base flows at different Reynolds numbers. Residuals are shown for the discrete u,v,w-momentum and pressure equations separately, and the continuous black line represents the theoretical convergence estimate for the Poisson equation of 0.125 per V-cycle. Each mark in the series represents a V-cycle of the multigrid solver with two linewise Gauss-Seidel relaxation sweeps on the downward leg and one sweep on the upward leg. The last computation on the right includes adaptive grid refinement: the finest grid covers only part of the computational domain covered by the coarser one.

Description of the baseflow

The steady-state flow around an infinite swept wing at Reynolds $Re_C = 10^6$ and sweep angle $\Lambda = 45°$ is presented in Figure 5.2. In the front view, the flow is coming towards the page, while in the top view the flow is moving from the bottom to the top. The pressure field in the (x, y)-plane is visualized using gray contour levels at the end of the front view and using colors on the wing surface. The maximum pressure of 0.5048 is obtained at the attachment line, while the minimum of -0.2401 is marked by the spanwise white line located about half way on the wing surface in the chordwise direction. Two sets of streamlines are shown, both originating upstream of the attachment line but with a small shift in the y-direction. The red one enters the boundary layer at a distance in y slightly above the height of the attachment line and stays at about one third of δ_{99} boundary-layer thickness. The gray one originates above the red and does not enter the boundary layer.

The main feature characterizing this flow field is a freestream velocity field which is not aligned with the pressure gradient: the velocity field has a non-zero spanwise z-component W while the pressure gradient is constrained to the (x, y)-plane because of the spanwise invariance of the solution. As showed in Figure 1.1 this misalignment results in curved streamlines, and the curvature is stronger in the boundary layer due to lower fluid momentum but unchanged pressure gradient. The additional streamline curvature in the boundary layer with respect to the inviscid flow results in a crossflow component illustrated in Figure 5.3.

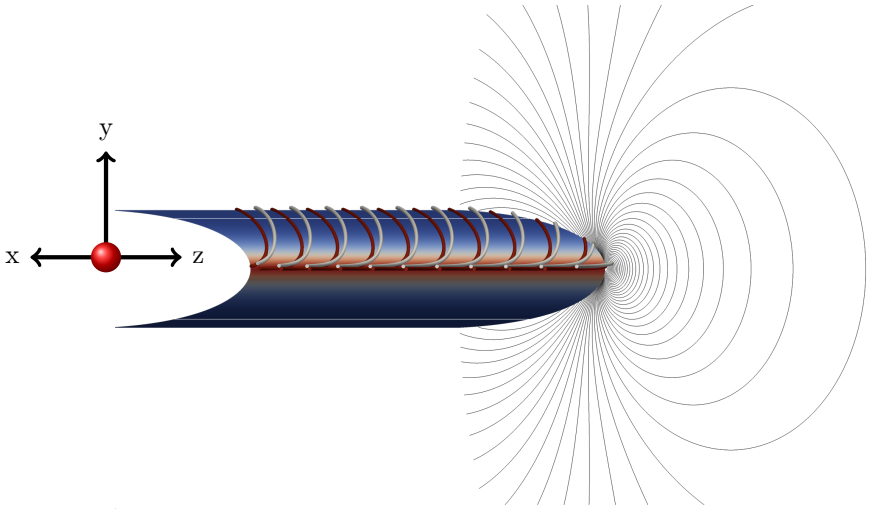
Following the path of a particle along streamlines in Figure 5.2, we can identify three areas characterized by changing signs of the pressure gradient. When approaching the leading edge, the U-velocity component of the flow — in the x-direction — is reduced by the adverse pressure gradient peaking at attachment line and extending well beyond the boundary-layer thickness while the spanwise W-component remains unchanged until the boundary layer effect becomes important. As a consequence, the streamlines are deflected in the spanwise direction. In particular, a streamline arriving at a height y corresponding to the leading edge will align itself with the leading edge and continue in the spanwise direction: a two-dimensional boundary layer spanning the entire wing is developed along the attachment line.

When moving along the airfoil past the pressure maximum at the attachment line, the effect of the pressure on the streamlines is opposite: a strong favorable pressure gradient — reflected in the concentration of pressure contours — increases the velocity component parallel to the solid boundary. As the spanwise component is not forced by any pressure gradient, it remains unchanged, and the streamlines are curved in the chordwise direction.

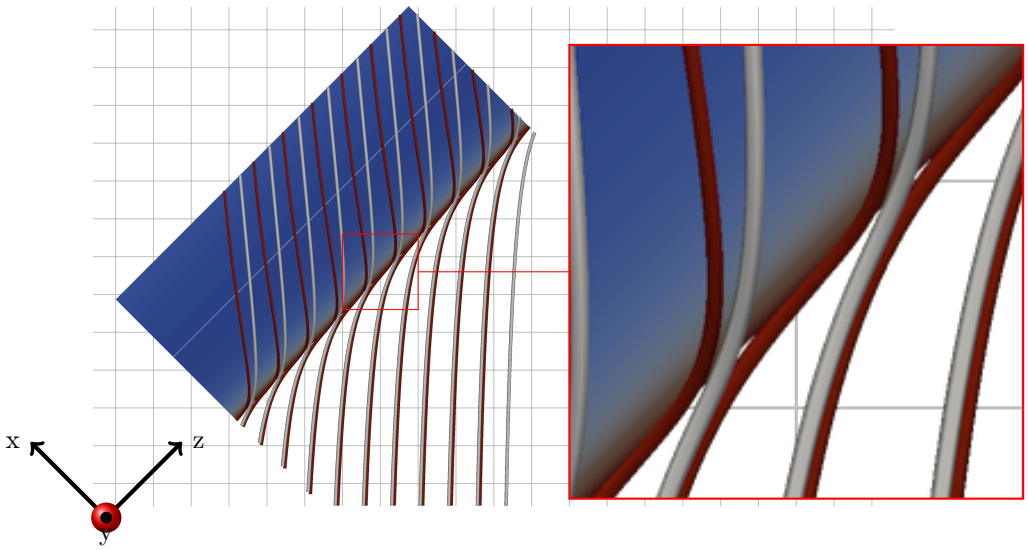
A third zone can be identified past the pressure minimum, where the pressure gradient is again adverse to the parallel velocity component. Accordingly, the streamlines are mildly curved back towards the inviscid flow direction.

A grid whose vertical lines are aligned with the inviscid flow $(U_\infty, 0, W_\infty)$ is shown in the top view to mark more clearly the streamline curvature.

Historically, these areas have been studied by using two different simplified models: (i) the swept Hiemenz flow [31, 15], describing flow impinging on an infinite plate at a given angle, has been used as a model of the region close to the attachment line, where the surface curvature is negligible, and attachment-line instabilities have been associated with viscous effects in the boundary layer; (ii) downstream of the attachment-line region a three-dimensional boundary layer model, characterized by a crossflow velocity component in the boundary layer as shown in Figure 5.3, has been used as a local model of the boundary layer, and inviscid crossflow instabilities have been associated with the inflection point in the crossflow velocity profile [58, and references therein].



(a) Front View



(b) Top View

Figure 5.2: Base flow streamlines. Both red and gray streamlines originate from the upstream zone: the red lines enter the boundary layer close to the attachment line while the gray lines cover the inviscid flow. The greater curvature of the boundary-layer streamline is due to the different magnitude of the gradient of the pressure close to the boundary as well as to the greater effect on the slow momentum fluid in the boundary layer. The solid boundary is colored with pressure values, and pressure contours are shown in black at one end of the wing. The white line parallel to the attachment line represents the position of the minimum of the pressure of about -0.24. The flow enters the page perpendicularly in front view and flows from the bottom to the top of the page in top view.

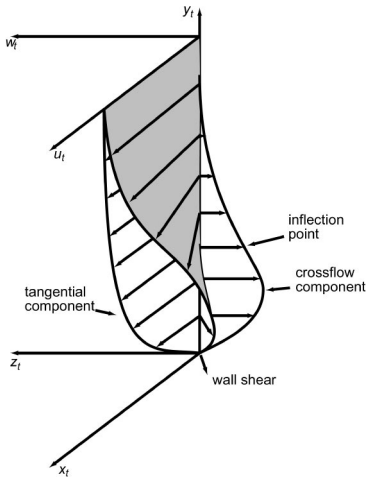


Figure 5.3: Velocity field within the boundary layer of a swept wing, from [58]. The x_t axis is aligned with the inviscid velocity, not with the chord. The swept wing flow is characterized by a freestream velocity field which is not aligned with the pressure gradient: the velocity field has a non-zero spanwise z-component W while the pressure gradient is constrained to the (x, y)-plane because of the spanwise invariance of the solution. This misalignment results in curved streamlines, and the curvature is stronger in the boundary layer due to lower fluid momentum but unchanged pressure gradient. The additional streamline curvature in the boundary layer with respect to the inviscid flow results in a crossflow component.

In this work, following the path first outlined by Mack et al. [44, 42, 43, 40], we consider a flow model including both the attachment-line boundary layer region as well as the boundary layer developing further downstream along the wing. This will allow us to address the stability problem from a global perspective.

To complete the description of the base flow, we present the δ_{99} boundary-layer thickness as a function of the Reynolds number; in addition, we report the pressure distribution and the δ_{99} boundary-layer thickness as a function of the chordwise coordinate s.

Figure 5.4 shows the boundary-layer thickness δ_{99}, based on the spanwise W-component of the velocity, measured at the attachment line, as a function of the chord-based Reynolds number Re_C for all the computations performed during the continuation process. Starting from a Reynolds number of about $Re_C = 10^5$ the expected relationship $\delta_{99} \propto Re_C^{-0.5}$ is obtained, and the proportionality constant is measured as 0.357.

In Figure 5.5 the computed pressure distribution at the solid boundary along the x-direction is plotted for three values of the Reynolds number (continuous lines) as well as for the inviscid case (dashed line). As noted at the beginning of the chapter, the computational domain for the base flow covers about 20% of the chordwise extent. The red inset shows that the pressure minimum decreases and moves towards the attachment line for increasing Reynolds numbers, and the solution for $Re_C = 10^6$, in blue, is nearly indistinguishable from the inviscid solution, represented by a black dashed line. The blue inset gives a detailed view of the outflow end of the numerical domain: as explained in chapter 3, the pressure is set to the inviscid solution at the outflow with a Dirichlet condition. The numerical boundary layer developed there is clearly seen in the $Re_C = 10^4$ curve which bends strongly over the last few mesh points.

Finally, Figure 5.6 shows the evolution of the boundary-layer thickness based on the spanwise W-component (δ_{99}, in blue) and the tangential component of velocity U_s (δ_{99}^U, in red) along the curvilinear

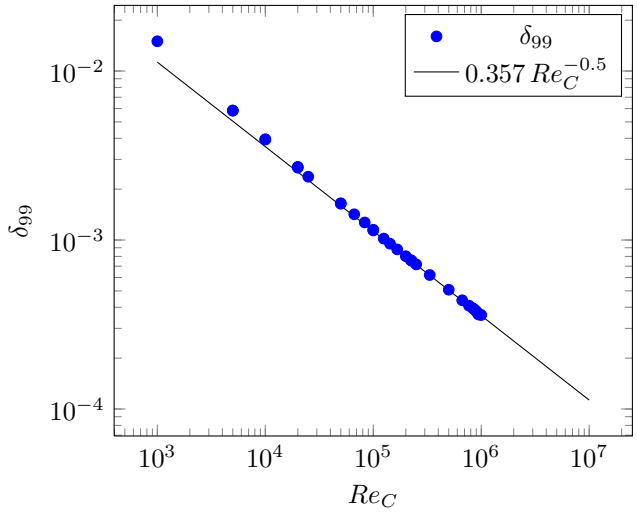


Figure 5.4: Boundary-layer thickness δ_{99} at the attachment line as a function of the Re_C number. The thickness δ_{99} is defined as the distance, measured normal to the wing-section profile, at which the spanwise velocity component w is 99% of its asymptotic value W_∞.

coordinate s. The W-based δ_{99} is always the thicker one and increases more slowly, in particular, close to the attachment-line region.

5.2 Global analysis of the direct operator

We now proceed to perform a global analysis of the perturbation problem ($\mathcal{L}$) using a modal approach to uncover the physical mechanisms governing the least stable mode.

The theoretical framework has been outlined in chapter 2. Appropriate use of the direct and adjoint modes allows us to shed some light on the central features of the perturbations: their structure, receptivity to forcing and sensitivity to modifications in the governing equations. Additionally, it is possible to identify and isolate the spatial location which is responsible for the development of self-sustained perturbations.

The numerical method employed is first introduced as a Krylov-subspace method with a shift-invert spectral transformation. The associated algorithms are already implemented in the SLEPc suite [29, 27, 30, 28]. SLEPc is extensively used in the present work for the solution of the discretized equivalent of both the direct and adjoint generalized eigenvalue problems defined in (2.14) and (2.24).

The computed part of the global spectrum is then presented. As already noted, it consists of eigenvalues with negative growth rates — corresponding to temporally decaying modes — as our numerical capabilities do not allow us to cross the critical Reynolds number in the base flow computation. We identify a branch of eigenvalues, composed of modes which are alternately symmetric and anti-symmetric (starting from the least stable mode and progressing towards the most stable). The spectrum has been recomputed for three domain sizes, characterized by different extents in the chordwise direction, which is equivalent to changing the location of the outflow boundary along the profile. It is shown that this change has no effect on the location of the spectrum. This result has previously been obtained in the case of the cylinder wake by Giannetti & Luchini [19]. As Giannetti & Luchini noted, this property is related to the fact that only the core, to be defined later, of the mode needs to be represented in the computation. We can thus anticipate this core to be located close to the attachment line, which is the

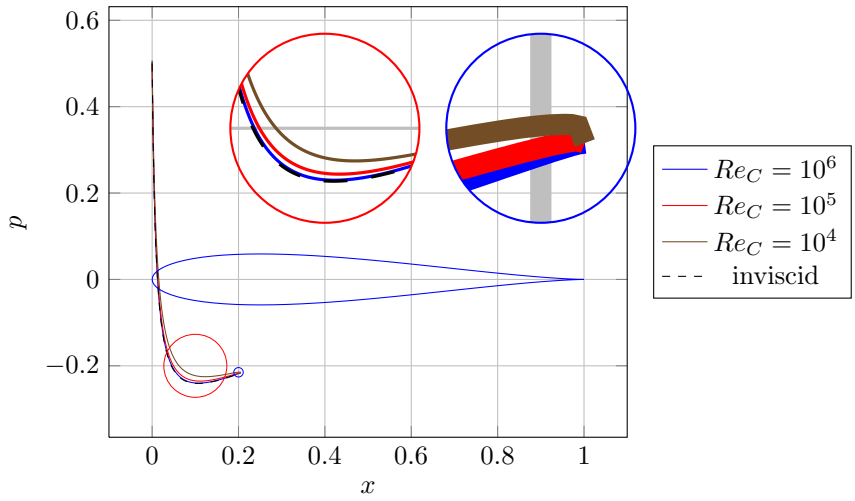


Figure 5.5: Pressure distribution at the wing surface for selected Reynolds numbers $Re_C = 10^4, 10^5, 10^6$. The analytically computed inviscid solution is represented as a black, dashed line. The pressure minimum moves upstream as the Reynolds number increases, and the solution for $Re_C = 10^6$ is nearly indistinguishable from the inviscid solution, as can be seen in the red inset. The effects of the Dirichlet pressure condition at the outflow is clear on the $Re_C = 10^4$ (brown) curve in the blue inset: the pressure is forced to be equal to the inviscid solution which results in a thin numerical boundary layer in the pressure field at the outflow. This area of the domain will be removed when the stability problem will be addressed.

only part consistently represented in all computations.

Next, the least stable direct eigenvector is analyzed. It is shown that it is consistent with both an attachment line mode [38, 39, 48, 49] and a crossflow mode [13] This feature has also been shown by Mack et al. [44] for the most unstable global mode of the compressible flow around a parabolic profile and gives us confidence in the fact that the main results of our analysis can be carried on to the unstable (supercritical) case. Again, the chordwise extension of the domain will be shown to be unimportant in determining the shape of the eigenvectors, provided that a small area in the vicinity of the leading edge is well represented.

Numerical procedure

A Krylov-Schur method with a shift-invert transformation is employed in order to numerically solve the generalized, non-Hermitian eigenvalue problem (2.14) and its corresponding adjoint (2.24). The chosen algorithm, among others, is implemented in SLEPc, "a software library for the solution of large scale sparse eigenvalue problems on parallel computers" [29, 27, 30, 28]. MUMPS [1, 2] is employed to perform the LU decomposition for the matrix inversion for the smaller of the three domains. Solution on the bigger domains have more stringent memory requirements, and a GMRES solver with an ILU preconditioner is employed in these cases. Both solvers (and many others) can be called from within the SLEPc library.

The complex plane is sampled by a series of different shifts for the shift-invert transformation. The shifts are located on the imaginary (phase speed) axis in order to cover the area between a phase speed of zero and one. The spanwise wavenumber selected for the following computations is $k_z = 4000$, corresponding to a wavelength which is about four times the boundary layer thickness close to the attachment line. While the choice of the wavenumber is quite arbitrary, as no "most unstable wavenumber" exists for our choice of Reynolds number and sweep angle, it corresponds to the dominant wavenumber observed

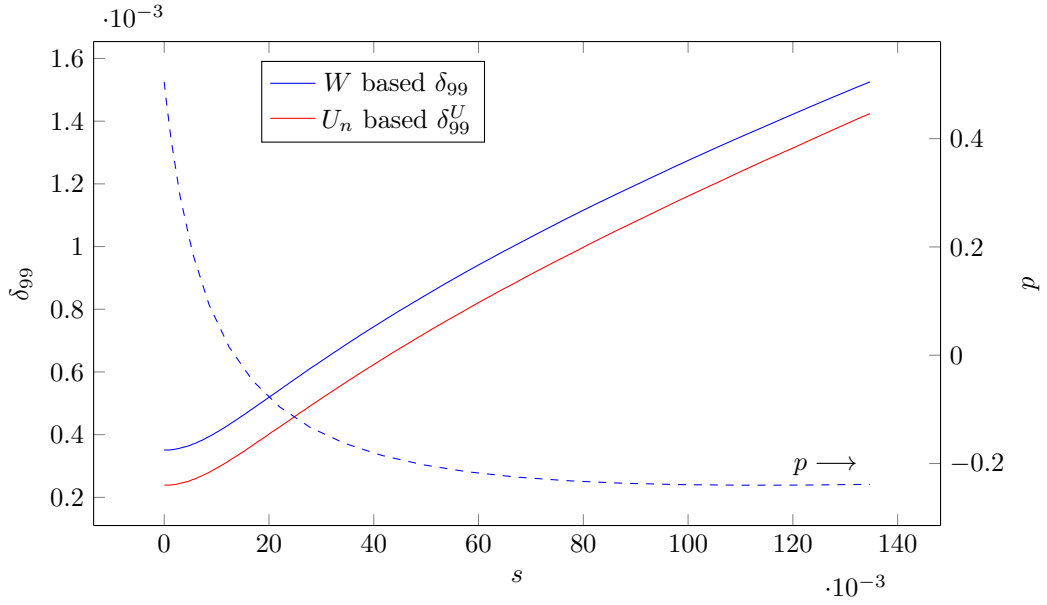


Figure 5.6: Boundary-layer thickness as a function of the curvilinear coordinate s. Both the spanwise-velocity-based δ_{99} and the tangential-velocity-based δ_{99}^U are shown.

in various experiments, as summarized by Dagenhart & Saric [13].

The development of the solver for the eigenvalue problem does not represent a major contribution of this work, and the reader is referred to the work of Mack [40] or the SLEPc technical reports [29] for a description of the employed algorithms.

Previous analyses

The modal approach to the investigation of the swept attachment-line boundary layer has previously been addressed by Lin & Malik [38, 39], Obrist & Schmid [48, 49], Mack, Schmid & Sesterhenn [44, 42, 43] and again by Obrist & Schmid [50].

Lin & Malik [38] used a Chebyshev spectral collocation method and regular polynomials to discretize the normal- and chord-wise direction of their domain in order to study the stability of the incompressible swept Hiemenez flow. They identified a branch of eigenvalues moving at approximately the same phase speed in the spanwise direction and showed that the most unstable mode was the symmetric Görtler-Hämmerlin mode, characterized by a linear dependence of the chordwise velocity component in the chordwise coordinate and an exponential decay outside the boundary layer. Less unstable modes were shown to alternate between antisymmetric and symmetric as one descends to smaller growth rates.

Obrist & Schmid [48] addressed the same problem by replacing the regular polynomials used by Lin & Malik in the chord-wise discretization with Hermite polynomials. They confirmed that the Görtler-Hämmerlin mode is the most unstable mode and identified a richer spectrum composed of several branches, continuous and discrete. Additionally, an analysis of non-modal effects and receptivity has been performed [49].

Mack, Schmid & Sesterhenn [44] and Mack & Schmid [40, 42, 43] addressed the stability of compressible flow around a swept parabolic body using a high-order finite-difference discretization scheme in both the normal and chordwise direction. They identified a global spectrum consisting of different branches: boundary layer modes, acoustic modes and wave-packet modes. Of these, only the boundary layer and

wave-packet branches are of interest for the current, incompressible study. Additionally, they showed, for the first time, evidence of a connection between attachment-line and crossflow instabilities. This result was made possible by considering a domain extending beyond the attachment-line region.

Global stability analysis

The computed spectrum of the linearized Navier-Stokes operator ($\mathcal{L}$) for Reynolds number $Re_C = 10^6$, sweep angle $\Lambda = 45°$ and spanwise wavenumber $k_z = 4000$ — corresponding to $Re_r = 16000$, $Re_s = 126$ and $\lambda/\delta_{99} \simeq 4$ — is shown with black symbols in Figure 5.7. It is made of a single branch of eigenvalues characterized by a nearly constant phase speed of 0.5, implying that the modes are travelling in the spanwise z-direction at half the velocity of the free stream. Inspection of the eigenvectors shows that symmetric ($S1$,$S2$,...) and antisymmetric ($A1$,$A2$,...) modes alternate when moving from the least stable eigenvalue to more stable ones. This result is consistent with the findings of Lin & Malik [38, 39] who, working in the unstable parameter range, identified a single branch at constant phase speed consisting of symmetric and antisymmetric modes.

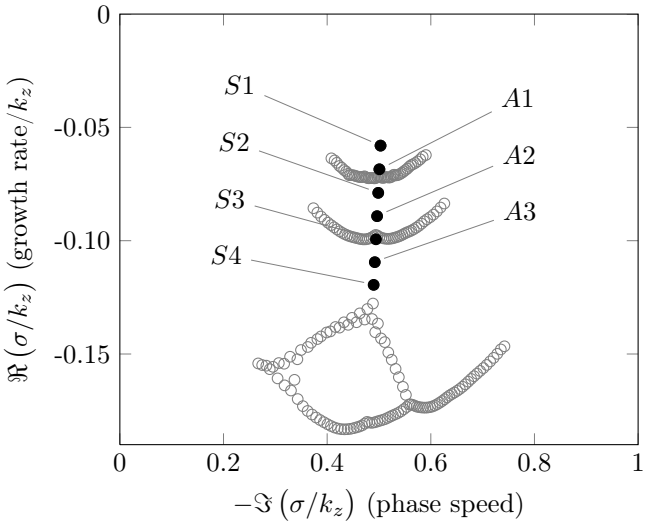


Figure 5.7: Eigenvalues for $Re_C = 1 \cdot 10^6$, $k_z = 4000$ ($\lambda/\delta_{99} \simeq 4$). The computed spectrum consists of a single branch — black, filled dots — of modes travelling at roughly the same phase speed $\Re\left(\sigma/k_z\right)$ of 0.5 in the spanwise direction. Symmetric ($S1$, $S2$, ...) and antisymmetric ($A1$, $A2$, ...) eigenvectors alternate when moving from the least stable to the most stable mode. The eigenvalues have been computed for three different domain sizes. For the large domain, only $S1$ and $A1$ are recovered. For the mid-sized domain the $S1, A1, S2, A2$ are recovered, and for the small domain all seven black eigenvalues are recovered. The gray, unfilled dots represent eigenvalues belonging to the pseudospectrum.

The spectrum has been computed for three domains differing in chordwise extent and is shown in Figure 5.7. A comparison of the different domains with the domain used for the base-flow computation and the full profile is displayed in Figure 5.8. The chordwise extent of the three domains is determined in numerical coordinates — shown in the leftmost part of Figure 3.2 — by the ranges $-0.75 \leqslant \xi \leqslant 0.75$, $-0.5 \leqslant \xi \leqslant 0.5$ and $-0.25 \leqslant \xi \leqslant 0.25$. The computation for the large domain returns only the $S1$ and $A1$ eigenvalues, together with the pseudospectrum represented by the curved branches in gray, unfilled dots right below $A1$. The mid-sized domain returns all four eigenvalues from $S1$ to $A2$ and its corresponding pseudospectrum is represented by the curved branches in gray, unfilled dots below $A2$. Finally, the

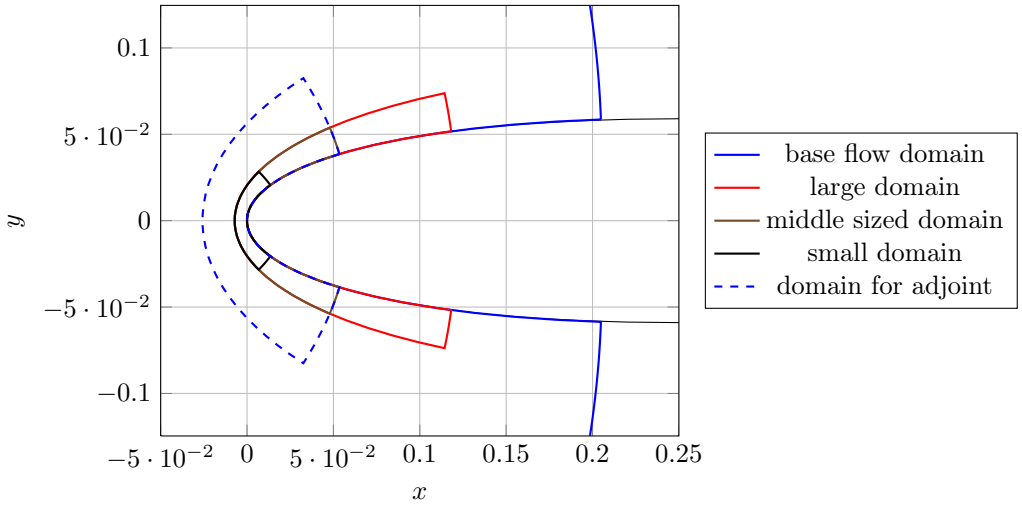


Figure 5.8: The three domain sizes (red, brown and black) used in the solution of the eigenvalue problem.

smallest domain returns all seven eigenvalues represented in Figure 5.7 and its pseudospectrum lies below $S4$ and takes a more complicated shape.

For comparison, the eigenvalues computed in the three domains are reported in table Table 5.1 with six decimal digits. The digits differing from the values obtained for the small domain are marked in red. It can be seen that the least stable eigenvalue $S1$ is the same in all three domains. The $A1$-mode has the same value for the small and mid-sized domain but the value obtained for the larger domain differs in the last four significant digits. The same is repeated for the $S2$- and $A2$-modes when comparing the small and the mid-sized domain: the $S2$-eigenvalue matches well while $A2$ differs in the last two significant digits. Computations on bigger domains return less eigenvalues since the increase in the number of degrees of freedom — required to mantain a constant mesh spacing — is not matched by an increase in the dimensionality of the Krylov subspace used in the eigenvalue computation: 100 vectors have been used for both the mid-sized and big domain despite the fact that the number of degrees of freedom increases by a factor of 1.5. In the small domain, 200 vectors have been used. Even if it may appear counterintuitive, the more precise results are expected to be the ones for the small domain where only a minor part of the flow structure is resolved. It is important to remark again that the same mesh spacing is used on all grids and the number of mesh points is increased when increasing the domain size.

We now move on to the description of the shape of the eigenvectors. The least stable mode $S1$ is visualized in Figure 5.9 using isosurfaces of the chordwise u-velocity component

$$u\left(x,y,z\right) = \Re\left(\hat{u}\left(x,y\right)e^{ik_z z}\right) \tag{5.1}$$

where $\Re$ denotes the real part, and the isosurface is at 10^{-10} of the maximum of u. Such a low contour level is required to visualize the eigenvector along the entire profile, as its magnitude changes over several orders of magnitude in the chordwise s-direction. The black surface represents the extension of the computational domain used in the base-flow computation corresponding to approximatively 20% of the chord length. The evolution of the "energy" of the eigenvector along the chordwise direction is presented in Figure 5.10. Three different regimes can be identified. Close to the attachment line, the isosurfaces are aligned with the chordwise direction. The $\hat{u}_s$-velocity component — tangential to the flow surface

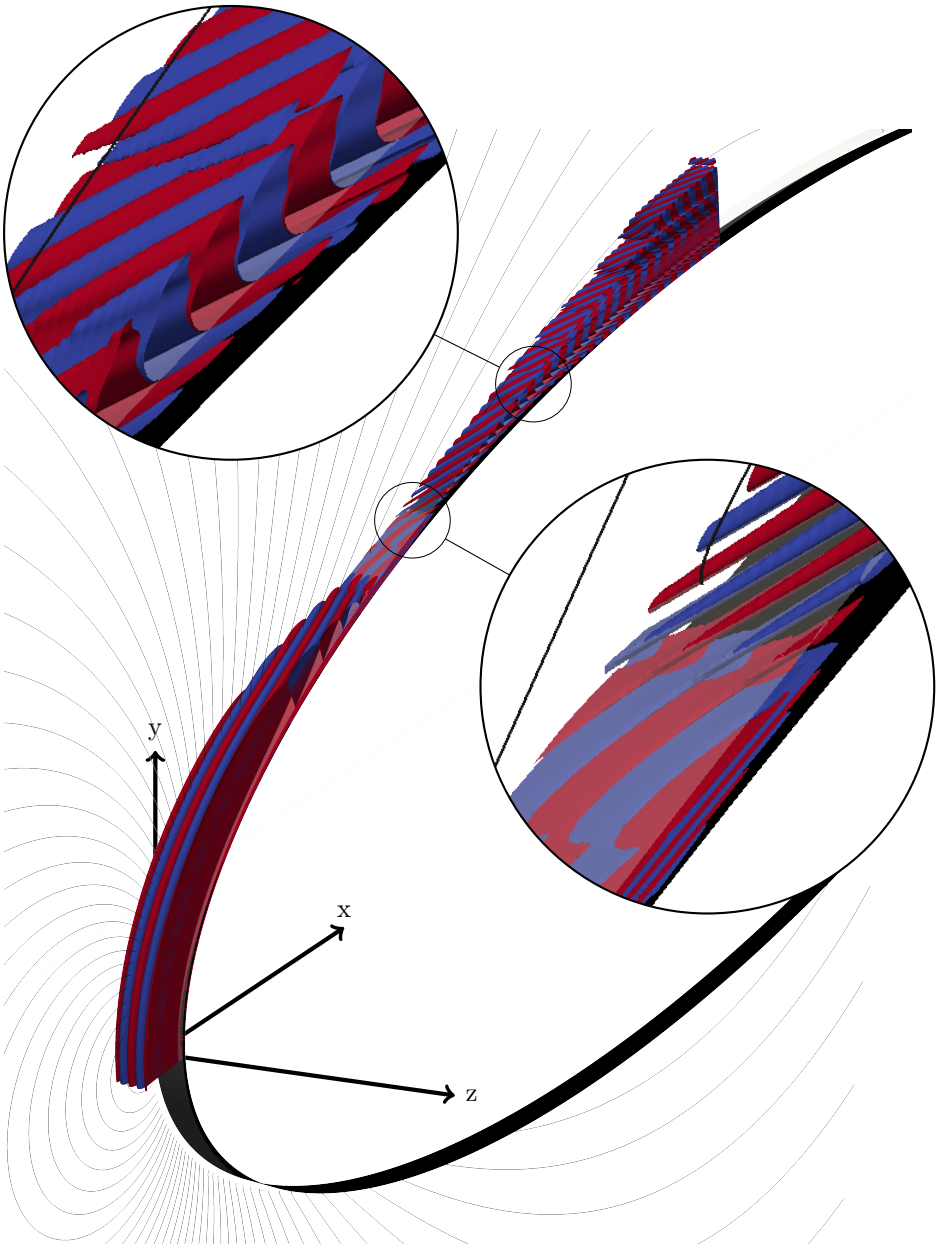


Figure 5.9: Eigenvector $S1$ represented by isocountour of the chordwise u-component of velocity at a contour level of 10^{-8} of the maximum value of the eigenvector. Only the upper half is represented. As already suggested by [26] and shown by Mack [44], the eigenvector displays features of both attachment-line modes close to the attachment line and crossflow modes further downstream. A more detailed view of the area where the "two modes" connect is provided in the lower circle: the isosurfaces first bend upstream, are compressed into the boundary layer and then realigned with the flow — the gray, transparent surface represents the δ_{99} boundary-layer thickness. Half way in the zoomed area, two isosurfaces co-exist, one over the other. The one on top, which lies at a height close to the δ_{99}-thickness, develops further downstream into the crossflow structures, whose maximum remains concentrated at the same height.

— increases linearly as can be seen in Figure 5.11 (blue curve), consistent with a Görtler-Hämmerlin mode which is known from swept Hiemenz flow studies to represent the most unstable global mode in the attachment-line region [38, 48].

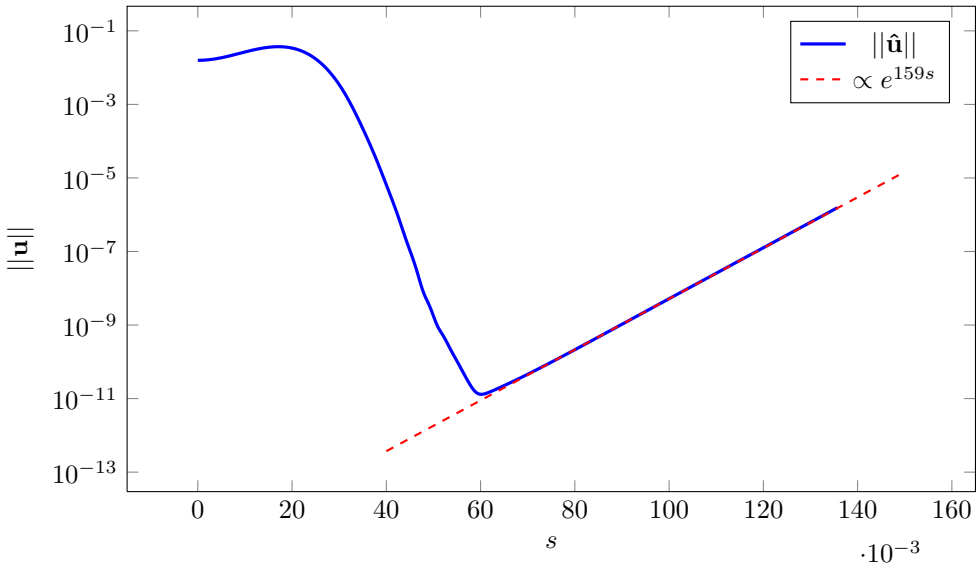


Figure 5.10: Norm of the velocity components of the $S1$-eigenvectors as a function of the curvilinear chordwise coordinate s. Semilogarithmic plot. The exponential growth of the crossflow structures is marked by the red dashed line, corresponding to the function $6.424 \cdot 10^{-16} \exp(159x)$.

Downstream of this region, the isocontours bend and align themselves in the direction transverse to the base flow. The energy of the mode along s decreases by more than ten orders of magnitudes with nearly exponential decay. In this transitional region, where the boundary-layer thickness increases steadily, the attachment-line structures transform into crossflow structures. The lower inset in Figure 5.9 shows this transition: crossflow structures, aligned with the external flow at 45° with respect to the chord, start to appear just below the δ_{99} boundary-layer thickness, represented by the semi-transparent surface. The crossflow structures then grow exponentially in the s-direction with a growth rate of 159, obtained by fitting the data of this area to an exponential curve (see the red dashed line in Figure 5.10).

Additionally, a section of the eigenvector in the (s, n)-plane at a fixed z-location is presented in Figure 5.12 in order to better illustrate its global shape. The attachment-line structure is clearly visible up to $3 \cdot 10^{-2}$, peaking in the n-direction at about one third of δ_{99} and exponentially decaying outside the boundary layer. The transitional region is between $3 \cdot 10^{-2}$ and $6 \cdot 10^{-2}$. Beyond this location, crossflow structures develop and grow exponentially in the s-direction.

In Figure 5.14 the same section is showed for the $A1$, $S2$ and $A2$ eigenvectors and Figure 5.13 present a comparison of the evolution of all eigenvectors in the chordwise direction.

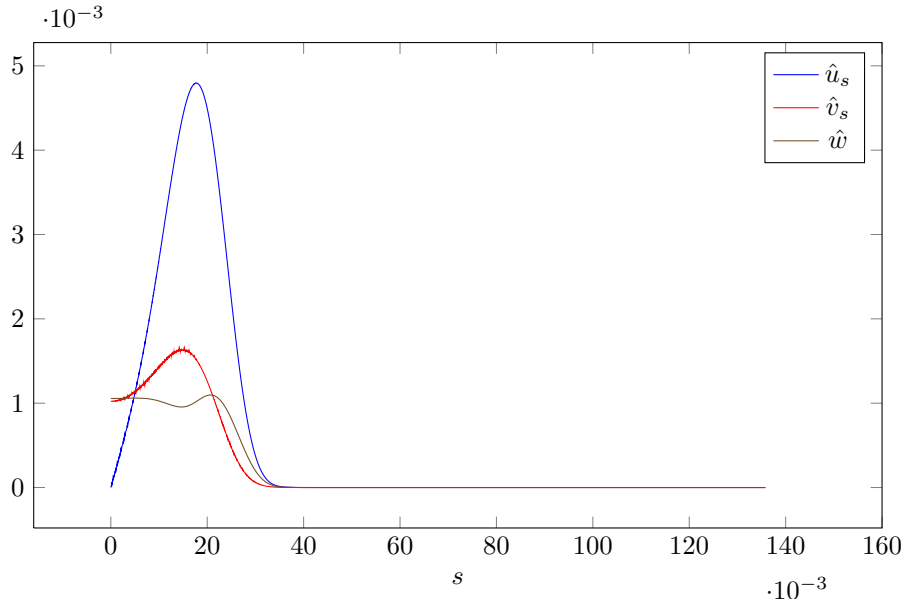


Figure 5.11: Velocity components of the $S1$-eigenvector at half the δ_{99} boundary-layer thickness as a function of the curvilinear chordwise coordinate s.

Table 5.1: Computed Eigenvalues

	Small domain $2049 \times 513 \times 3^{\dagger}$ — $200^{\ddagger}$	Middle sized domain $4097 \times 513 \times 3^{\dagger}$ — $100^{\ddagger}$	Large domain $6145 \times 513 \times 3^{\dagger}$ — $100^{\ddagger}$
$S1$	$-232.212795 - 2012.093989\,i$	$-232.212795 - 2012.093989\,i$	$-232.212795 - 2012.093989\,i$
$A1$	$-274.036727 - 2003.162782\,i$	$-274.036727 - 2003.162782\,i$	$-274.031158 - 2003.163902\,i$
$S2$	$-315.544891 - 1994.259873\,i$	$-315.544891 - 1994.259873\,i$	—
$A2$	$-356.710140 - 1985.385797\,i$	$-356.710150 - 1985.385780\,i$	—
$S3$	$-397.505318 - 1976.541365\,i$	—	—
$A3$	$-437.903326 - 1967.724210\,i$	—	—
$S4$	$-477.978550 - 1958.896547\,i$	—	—

Marked in red are the digits that change with the domain size. For each eigenvalue, digits in black are the same for all domain sizes.
† Degrees of freedom in the chordwise and normal direction and number of unknowns for each grid point;
‡ Size of the Krylov subspace used in the computation;

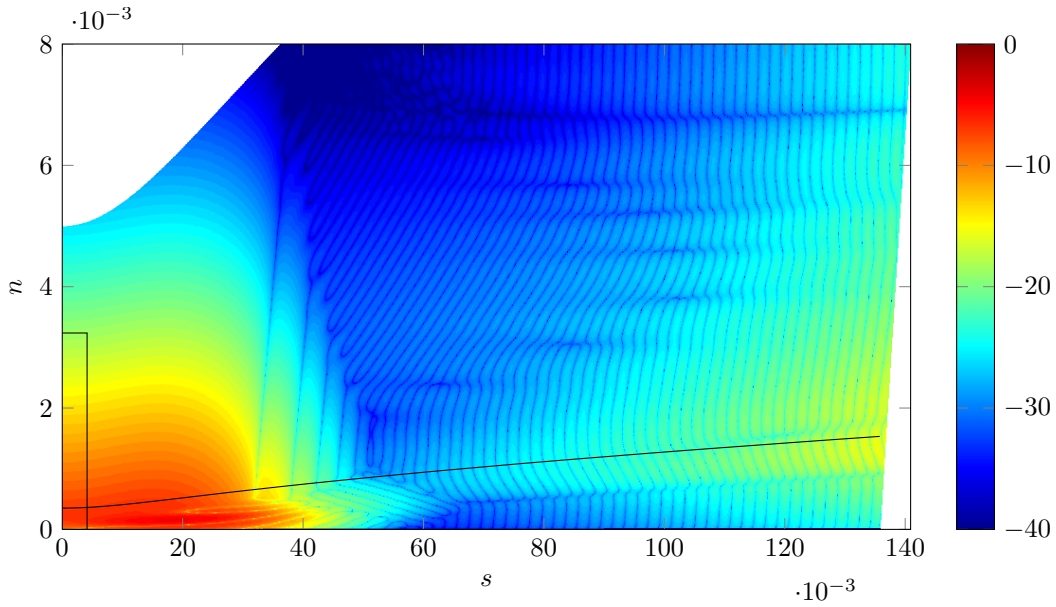


Figure 5.12: Eigenvector $S1$ in the (s, n)-plane, visualized by the $\Re(\hat{u})$-velocity component, logarithmic scale. The attachment-line structure is clearly visible up to $3 \cdot 10^{-2}$, peaking in the n-direction at about one third of δ_{99} and exponentially decaying outside the boundary layer. The transitional region is between $3 \cdot 10^{-2}$ and $6 \cdot 10^{-2}$. Beyond this location, crossflow structures develop and grow exponentially in the s-direction. Only positive values of s are shown, and the s-axis is compressed to visualize the full chordwise extent. The δ_{99} boundary-layer thickness is represented by a black line. The eigenvector is symmetric with respect to the n-axis. The rectangle close to the n-axis will be later used to visualize the adjoint eigenvector and the wavemaker.

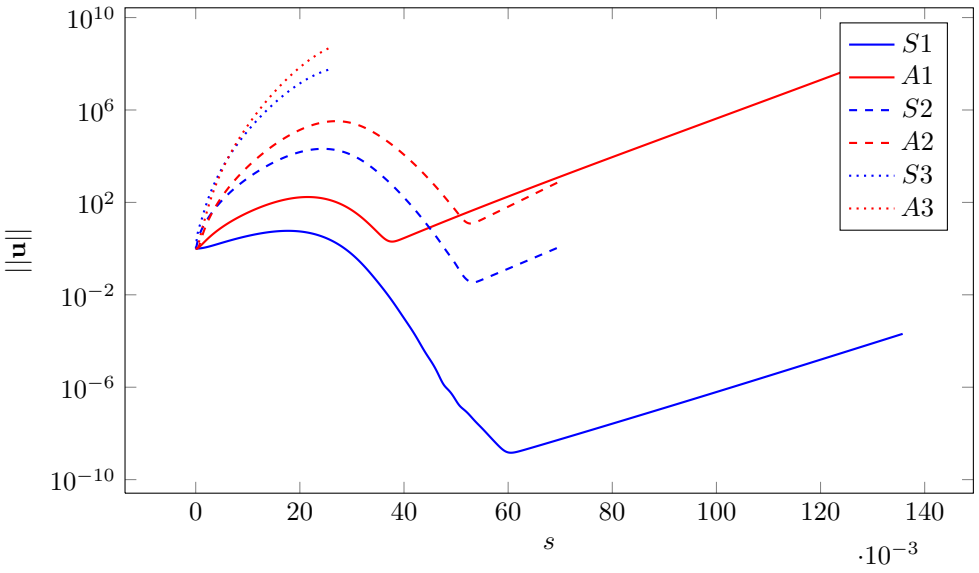


Figure 5.13: Norm of the in-plane velocity components (u, v) of the first six eigenvectors as a function of the chordwise coordinate s. Eigenvectors in this figure are normalized such that their magnitude is one at attachment line. The first four eigenvectors clearly show the attachment-line structure and the exponentially growing crossflow structures. For the two more damped eigenvectors (dotted lines) the computational domain is truncated before the beginning of the crossflow structures. From this plot it is clear that the peak of the attachment-line structure moves downstream when descending to smaller growth rates.

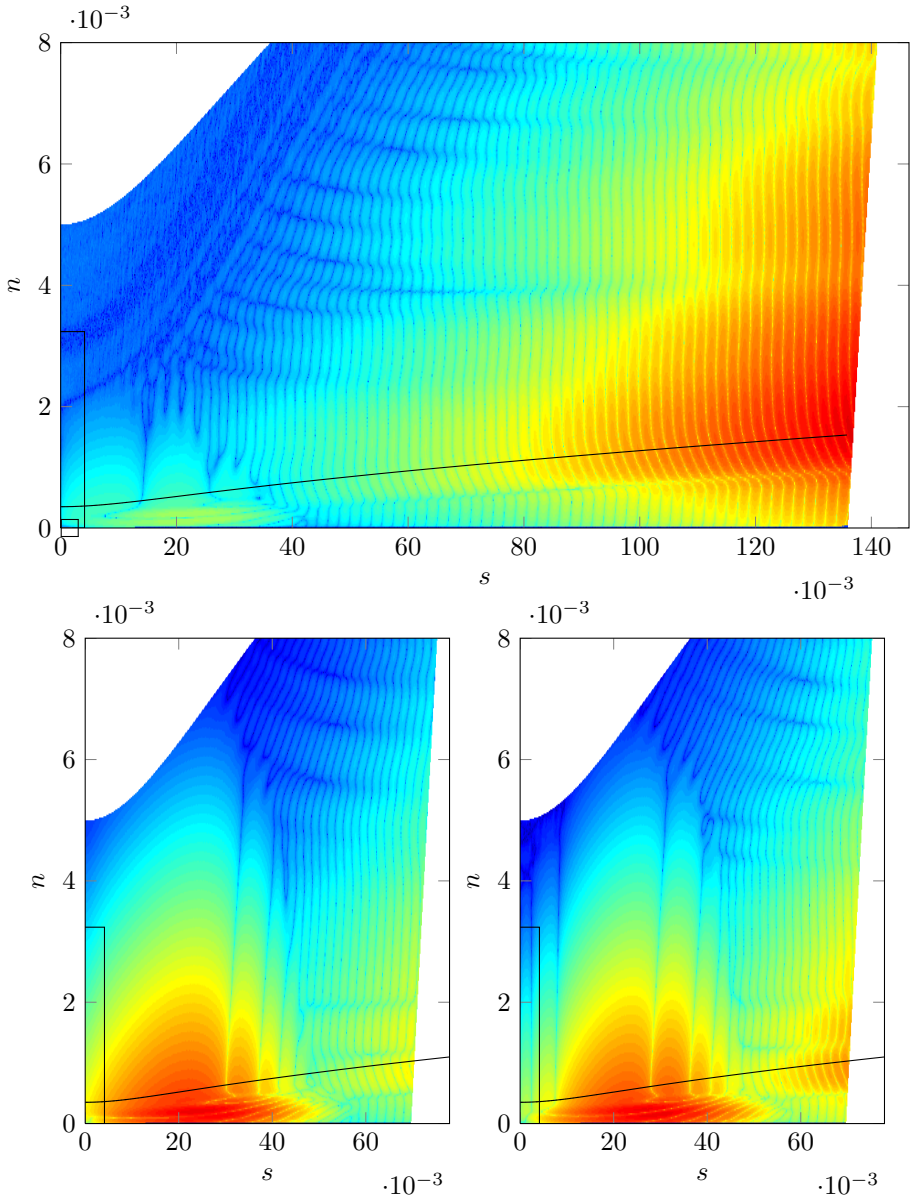


Figure 5.14: Eigenvectors $A1$ (top), $S2$ (bottom left) and $A2$ (bottom right) in the (s, n)-plane, visualized by the $\Re(\hat{u})$-velocity component, logarithmic scale. The computational domain used for the $S2$ and the $A2$ eigenvectors has a chordwise extent equivalent to approximately half the computational domain used for the $S1$ and $A1$ eigenvectors. In all eigenvectors, an attachment-line structure is clearly visible, followed by a transition region and crossflow vortices structures growing exponentially in the s-direction. The peak of the attachment-line structure moves downstream when descending to smaller growth rates.

5.3 Adjoint field

As has been shown in section 2.4, the adjoint field represents the receptivity of any scalar objective functional to a forcing of the perturbation equations or to a structural modification of the operator $\mathcal{L}$. Recalling equations (2.26) and (2.29), we can write the variation of a generic objective functional obj as a function of a variation in the forcing $\delta\hat{\mathbf{f}}'$ or in the operator δA as

$$\delta\left(obj\right) = -\left\langle \hat{\mathbf{q}}^{+}, \delta\hat{\mathbf{f}}' \right\rangle_{\Omega} \tag{5.2}$$

$$\delta\left(obj\right) = \left\langle \mathbf{q}^{+}, \delta A\, \mathbf{q} \right\rangle_{\Omega} \tag{5.3}$$

In a sense, the adjoint field projects a variation in the forcing or in the operator into the direction of the objective functional.

The same procedure used in solving the direct eigenvalue problem is employed for the adjoint. The discretization of the adjoint governing problem is obtained by computing the complex conjugate transpose of the discretized direct problem, as we have seen at the end of section 2.3. Consequently, the adjoint spectrum is the complex conjugate of the spectrum of the direct operator shown in Figure 5.7: the only difference is in the sign of the phase speed. As is the case of the direct problem, the domain size does not influence the results.

When computing the complex conjugate transpose, particular attention has to be paid to the boundary conditions to limit the introduction of numerical errors. To demonstrate what kind of difficulties can arise, we consider a discretization of the one-dimensional Laplacian operator on a five-point grid of mesh size h. A Dirichlet boundary condition is implemented at the left edge of the domain by setting the diagonal value of the first line of the matrix to one. A Neumann boundary condition is implemented at the right edge and is discretized by a first-order finite-difference discretization in the last line of the matrix. The discretized matrix and its adjoint read

$$\begin{bmatrix} 1 & & & & & \\ 1/h^2 & -2/h^2 & 1/h^2 & & & \\ & 1/h^2 & -2/h^2 & 1/h^2 & & \\ & & 1/h^2 & -2/h^2 & 1/h^2 & \\ & & & 1/h & -1/h & \end{bmatrix}^{H} = \begin{bmatrix} 1 & 1/h^2 & & & & \\ & -2/h^2 & 1/h^2 & & & \\ & 1/h^2 & -2/h^2 & 1/h^2 & & \\ & & 1/h^2 & -2/h^2 & 1/h & \\ & & & 1/h^2 & -1/h & \end{bmatrix} \tag{5.4}$$

where marked in red are the only elements that change when the adjoint is computed. If we consider the first and the last line of the adjoint matrix, it is clear that the value on the boundary will be scaled by a factor $1/h^2$ and $1/h$ for the left and right edge of the domain, respectively, resulting in large values on the boundary. This effect can be avoided by considering the fact that the eigenvalue system is homogeneous: the boundary condition equations in the direct equations can be arbitrarily multiplied by a constant — or the Laplacian operator can be rescaled by multiplying by h^2 to have coefficients of $\mathcal{O}\left(1\right)$. Numerical difficulties related to the eigenvector normalization during the solution process can thus be avoided, and the equivalent two-dimensional approach is used in our computations. It should be noted that while we can solve the numerical problem, the fact remains that the values of the discrete adjoint field on the boundary are undetermined.

The least stable $S1$ adjoint eigenvector is visualized in Figure 5.15 using isosurfaces of the adjoint u-field computed as in equation (5.1). As in the visualization of the direct eigenvector in Figure 5.9, the black surface represents the extent of the computational domain used in the base-flow computation,

corresponding to approximatively 20% of the chord length. The adjoint field covers an area close to the attachment-line region and upstream of it, indicating that any forcing or structural modification (for example a change in the base flow) outside this region has no or little influence on the objective functional and is ineffective in trying to control the behavior of the $S1$-mode. The much larger contribution of the adjoint field is located inside the boundary layer and extends only a few boundary-layer thicknesses δ_{99} in the chordwise direction across the attachment line, as can be seen in Figure 5.17. This spatial extent corresponds to about one hundredth of the full s-extent of the computational domain used for the base flow; it identifies the area of the domain in which forcing has to be applied in order to control the $S1$-mode of the flow.

Comparison of the direct and the adjoint eigenvector clearly demonstrates the degree of the operators' non-normality described in section 2.4: as is the case for the flow around a cylinder, the two eigenvectors mostly cover different parts of the domain and overlap only in a small region which, for the wing profile, is close to the attachment line.

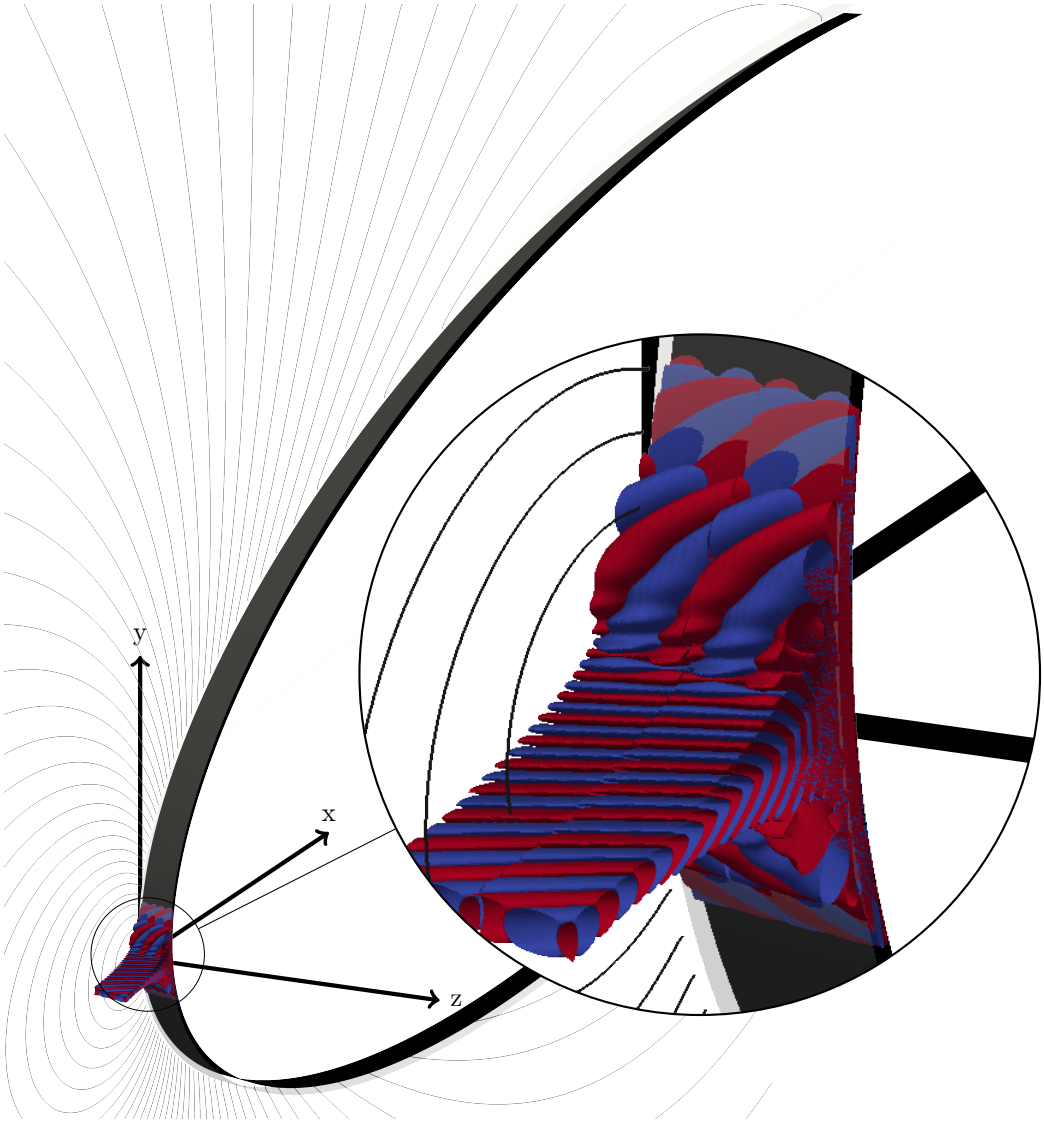


Figure 5.15: Adjoint eigenvector $S1$ represented by isocountour of the chordwise u-component of velocity at a contour level of 10^{-6} of the maximum value of the eigenvector.

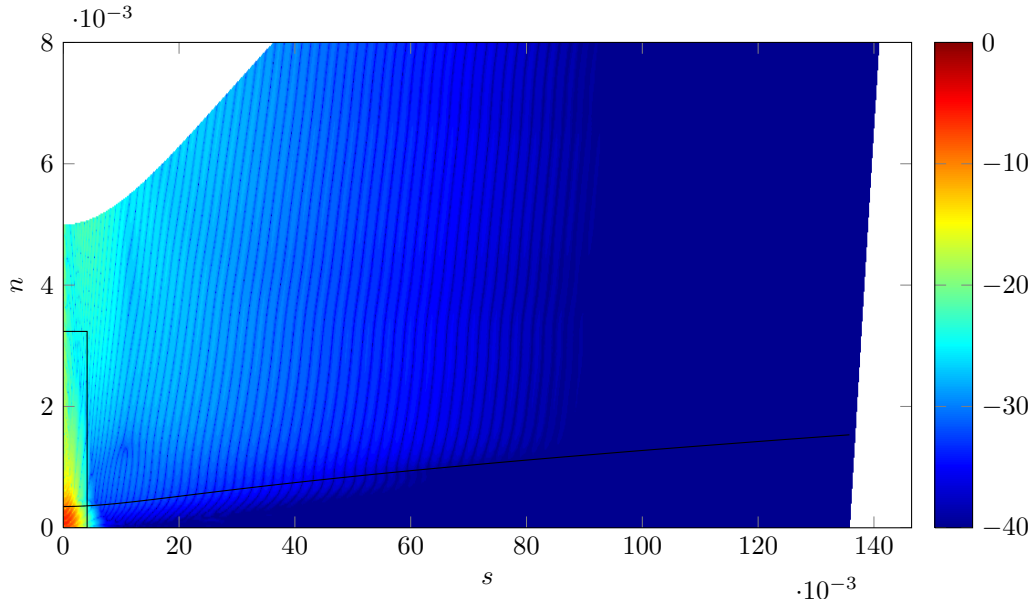


Figure 5.16: Adjoint eigenvector $S1$ in the (s, n)-plane, visualized by means of the $\Re(u)$-component, logarithmic scale. The adjoint eigenvector decays exponentially and monotonically in the chordwise direction and its maximum is located close to attachment line.Only positive values of s are shown, and the s-axis is compressed to visualize the full chordwise extent. The extension of the computational domain is equivalent to the one used for the corresponding direct eigenvector in Figure 5.12. The δ_{99} boundary-layer thickness is represented by a black line. The black rectangle in the bottom-left corner marks the extension of the domain represented in Figure 5.17.

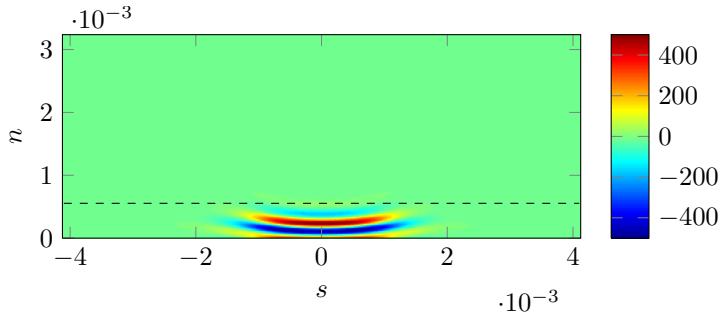


Figure 5.17: Adjoint field close to the attachment line. Significant contributions of the adjoint field are located inside the boundary layer and extend only a few boundary-layer thicknesses δ_{99} in the chordwise direction across the attachment line. This length corresponds to about one hundredth of the full s-extent of the computational domain used for the base flow. It localizes the region of the computational domain where forcing is most effective in controlling the flow. The normalization condition for the adjoint field is given by equation (2.28).

5.4 The wavemaker

The feedback mechanism of the perturbations described by Giannetti & Luchini and introduced at the end of chapter 2 results in the identification of the area of localized, pointwise feedback as the pointwise product of the direct and adjoint eigenvectors, described by equation

$$\lambda(\mathbf{x}) = -\mathbf{q}^{+}(\mathbf{x})\ \mathbf{C}_0\,\mathbf{q}(\mathbf{x})\,.$$

The wavemaker for the least stable $S1$ adjoint eigenvector is visualized in Figure 5.18 using isosurfaces of the u-component of $\lambda(\mathbf{x})$. Isosurfaces are at 10^{-8} of the maximum of u. Analogous to the visualization of the direct and adjoint eigenvector, the black surface represents the chordwise extent of the computational domain used in the base-flow computation — see Figure 5.8 for a comparison with the computational domains used for the eigenvalue problems.

As predicted by the previous analysis of the direct and adjoint eigenvectors, the maximum of $\lambda(\mathbf{x})$ is close to the attachment line. Figure 5.19 presents a detailed view of the same area, used for visualization of the adjoint eigenvector in Figure 5.17. The fact that the direct $S1$-eigenvector varies very slowly in the chordwise direction close to the attachment-line area results in the similarity between the adjoint field and the $\lambda(\mathbf{x})$-function.

We now return to the previous observation on the invariance of the results with respect to the domain size and, as noted by Giannetti & Luchini [19], define the region of the flow, which governs the behavior of the $S1$-mode, as the region where $\lambda(\mathbf{x})$ attains its maximum. We can then try to recompute the spectrum on a domain including only the area represented in Figure 5.19, as λ is nearly zero anywhere else. Maintaining the same mesh spacing used in the larger domains, this area corresponds to a 400×200 mesh-point domain, which is extremely small when compared to even the smaller domain previously used at 2049×513 mesh points. The new eigenproblem becomes easily solvable using Matlab's `eigs`-function. The resulting spectrum is shown in Figure 5.20. Even with this small domain size, the first four eigenvalues closely match the eigenvalues computed on larger domains. Additionally, a larger part of the spectrum can be uncovered, first because of the reduced ratio between the number of degrees of freedom and the number of vectors used in building the Krylov subspace and, second, due to the reduced computational cost and the possibility of increasing the dimensionality of the Krylov subspace. The spectrum is composed of various branches which will require further investigation to offer a physical explanation. It should be noted that the fact that the wavemaker is concentrated in the attachment line is a characteristic of the first eigenvector and of the other modes within the same branch; it must not be taken as a general property of the complete spectrum, however. Nonetheless, branches similar in shape and location to the ones showed in Figure 5.20 have been obtained for larger domains but lower wavenumbers and are worth being investigated in a future effort.

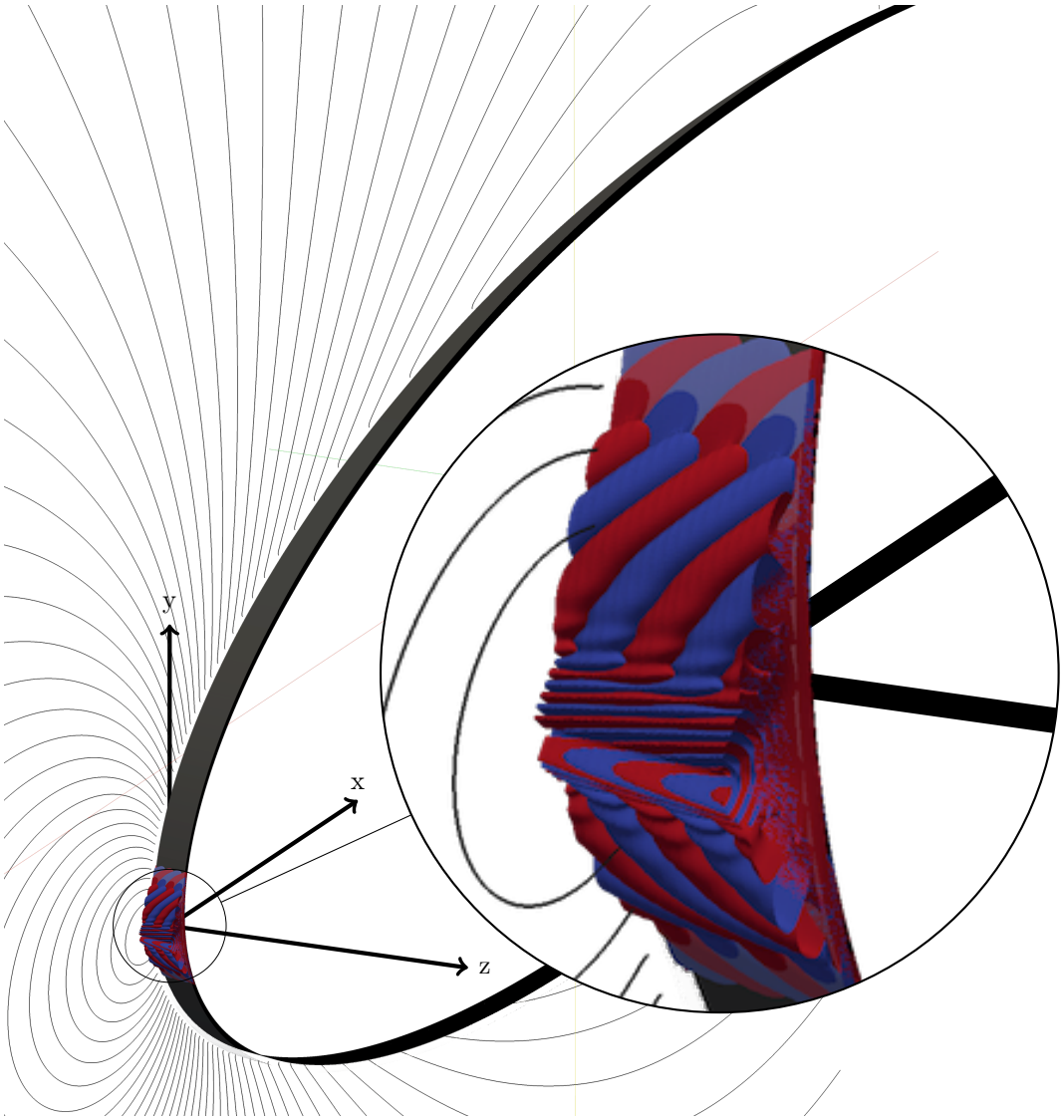


Figure 5.18: $\Re(u)$-component of $\lambda(\mathbf{x})$ for the $S1$ mode.

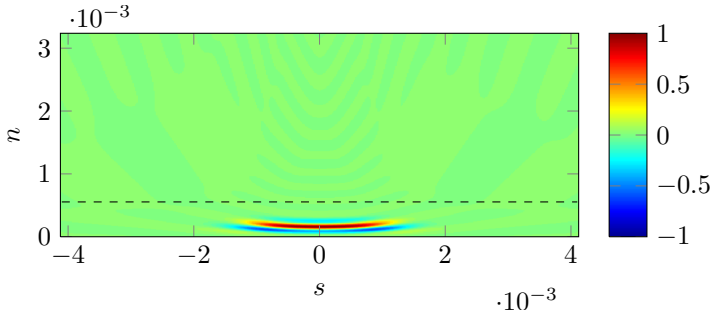


Figure 5.19: The function $\lambda(\mathbf{x})$ for the $S1$-mode, visualized by means of the $\Re(u)$-component. The size of the domain represented is the same as for the adjoint in Figure 5.17 and the λ-function closely resembles the adjoint vector due to the very slow chordwise variation of the associated direct $S1$-mode near the attachment line.

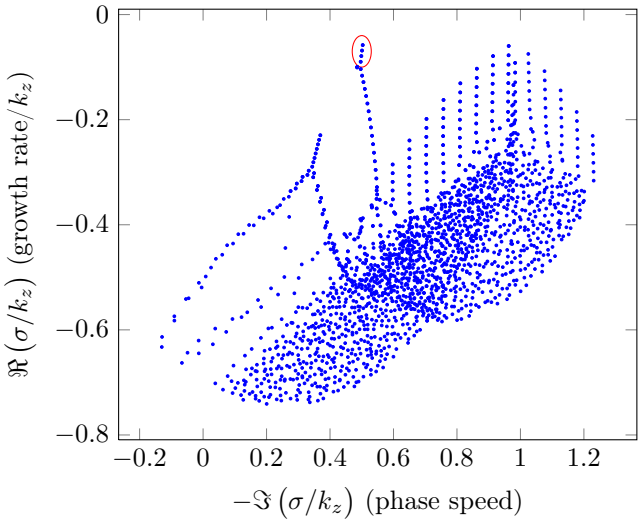


Figure 5.20: Computed spectrum for a domain containing only the wavemaker. Circled in red are the eigenvalues corresponding to the four least stable eigenvalues of Figure 5.7

Chapter 6

Conclusions and Perspectives

Two paths have been explored during the four years spent on this project. The first, mainly numerical, resulted in the development of a multigrid solver capable of directly computing the steady-state solution of the nonlinear Navier-Stokes equations. The second, more physical, is the determination of the receptivity and sensitivity properties of the flow characterizing the three-dimensional boundary layer forming in the attachment-line region of a swept wing.

The analysis of the multigrid framework performed in chapter 4 clarifies the main difficulties that had to be overcome in order to obtain the multigrid efficiency predicted by theory. The analysis presented and the code developed is heavily based on previous work and, in particular, on the results of Diskin [14] and Swanson [64]. Their analysis has been fundamental in overcoming some of the difficulties encountered during my work. Nonetheless, the development of a multigrid solver showing theoretical (or nearly theoretical) efficiency at such high-Reynolds-number viscous flow is, to my knowledge, novel. Grid stretching and adaptive grid refinement are additional, useful features of the developed code. The multigrid solver has been successfully used to compute the base flow around the leading-edge region of a swept wing.

Despite being the subject of much research, due to both its academic and industrial interest, the swept-wing problem has not been completely solved, and the exact mechanisms governing the transition from laminar to turbulent flow are still the subject of active research.

The focus of this work has been put on receptivity to forcing and sensitivity to structural perturbations of the operator, and the identification of the most receptive and the most sensitive regions for the least stable eigenvector is the main contribution of this work. An eigenvalue/eigenvector approach has been used to describe the dynamical system governing the evolution of the perturbations in order to extract the coherent structures describing the intrinsic flow behavior. As shown by Giannetti & Luchini [19] and Marquet, Sipp & Jacquin [45] for the cylinder wake, in this approach receptivity provides the variation of the amplitude of the eigenvectors as a function of the variation of the forcing, while sensitivity provides the shift of the eigenvalues in the complex plane as a function of the changes in the governing operator. The adjoint field has been shown to be at the center of the definition of both receptivity and sensitivity by using a Lagrangian approach. In this approach the governing equations are implemented as constraints by means of Lagrangian multipliers, and stationary points of the Lagrangian have been sought which recover the governing equations and define an associated adjoint problem.

The least stable eigenvector, which is expected to resemble the most unstable mode as the critical Reynolds number is crossed, shows structures characteristic of both attachment-line [25, 38, 39, 48, 49] and crossflow [13] instabilities. This coexistence of different structures in the same eigenvector had been already suggested by Hall [26] and lately observed by Mack et al. [44] for a compressible attachment-line configuration in an unstable parameter range. It has been analyzed in more detail in this work. The eigenvector shows a variation in magnitude of several order of magnitude along the chordwise direction: an initial growth close to the attachment line, characteristic of the Görtler-Hämmerlin mode, is followed by a drastic decrease of roughly ten orders of magnitude ultimately leading to the transition from

attachment-line to crossflow structures. Downstream of the transition region, the crossflow structures grow exponentially in the chordwise direction.

This direct eigenvector spans the entire chordwise extent of the domain and is still exponentially growing as the domain's outflow boundary, located close but upstream of the pressure minimum, is reached. In contrast the adjoint eigenvector is localized in a rather small area extending only a few boundary-layer thicknesses across the attachment line. Receptivity to forcing is accordingly localized in the same area, providing a strong indication where in the flow it is most effective to apply a control strategy. Forcing elsewhere in the domain would have very little effect on the amplitude of the eigenvector.

A similar observation holds for the sensitivity: the location of the wavemaker is mostly determined by the location of the adjoint eigenvector, again identifying a region a few boundary-layer thicknesses across the attachment line as most responsive for the structure and evolution of the least stable eigenvector. The result already obtained by Giannetti & Luchini [19] for the cylinder wake is confirmed for the swept-wing boundary layer: to identify the correct eigenvalue, only the area containing the corresponding wavemaker needs to be represented in the computations. This suggests that results from previous analyses based on swept Hiemenz flow should be essentially correct, as the wavemaker area is so confined that it can be well approximated by this simplified Hiemenz flow model.

Perspectives and future work

Two paths have been explored in this work, and two paths are suggested for future efforts.

Multigrid has been proven to be a very effective approach in solving the numerical problem arising from the discretization of the Navier-Stokes equations. The solution of the linearized Navier-Stokes equations, not shown in this work, is even less expensive. Extension of the solver to complex-valued problems would allow the efficient solution of the linear, complex problem involved in the construction of the Krylov subspace during the eigenvalue computations.

A limitation of this work has been the inability of applying adaptive grid refinement in the eigenvalue solver. As a consequence, stretched grids had to be introduced. Adaptive grid refinement itself had not been used to its full power in the computation of the base flow. With a complex-valued multigrid solver available, adaptive grid refinement could be easily implemented in the eigenvalue solver as well: the Krylov subspace vectors would be defined on a single composite grid made of all mesh points that do not have a finer-grid representation, and all standard routines used in a single-grid solver could be employed. This composite grid would then be scattered to all grids used in the multigrid solver when a solution of the linear solver is required, and the so-computed solution would be scattered back to the composite grid to obtain the new vector of the Krylov subspace. A more intriguing, but possibly more complex, possibility is to treat the eigenvalue problem as a nonlinear problem using the Full Approximation Scheme (FAS). This latter idea dates back to 1983 [11] but to my knowledge few applications have been tested [10]. Brandt (personal communication) suggested the work of Kushnir on Data Analysis [37] as a good starting point for developing an eigenvalue multigrid solver for the linearized Navier-Stokes equations. A review of the possibilities offered by multigrid, including in the field of control, is given by Brandt in [9].

More work has also to be done concerning the physics of swept attachment-line boundary layer. A more in-depth analysis of the results obtained during this work is recommended to uncover the whole significance of the adjoint field and of the wavemaker for the possibility of effectively controlling the flow. The approach used by Marquet, Sipp & Jacquin [45] to study the cylinder wake is similar to what has been outlined in this work and can be used as a starting point for a future analysis.

Almost overlooked in this work is the question of the meaning of boundary values of the adjoint field.

As noted in section 5.3, obtaining the adjoint operator by computing the complex-conjugate (Hermitian) transpose of the direct operator does not uniquely determine the magnitude of the adjoint field on the boundary. More interestingly, Giannetti & Luchini [19, end of section 4] make a distinction between receptivity to forcing and receptivity to initial conditions, thus justifying the fact that the left eigenvector associated with the linearized Navier-Stokes operator is not zero on the boundary and that the pressure component of the adjoint field is identically zero, however. This observation could have interesting consequences on the analysis of the effect of forcing at the wall (in contrast to forcing very close to the wall).

A parametric study is also suggested. First, the Reynolds number should be increased until an unstable configuration is reached. This would be useful also in light of the observation on the small extent of the wavemaker: a critical Reynolds number, matching the one computed for the swept Hiemenz model [25, 38, 39, 48], would provide additional confirmation of the fact that even this simplified model can correctly predict the stability behavior of most of the boundary layer on swept wings. Secondly, the dependence of the critical Reynolds number on the spanwise wavenumber should to be assessed and compared to previous results in the unstable regime. Third, an analysis of effects of sweep angle variations could provide more insight into the coupled behavior of Tollmien-Schlichting waves and crossflow vortices.

Some thoughts against writing your own research code

During this work, a substantial effort has been put into developing the multigrid solver used to compute the base flow. The original idea was to develop a pressure-correction based time-stepper solver, building on previous work by Mack [41], and using a multigrid solver to compute solutions to the Poisson equation for the pressure appearing in the context of fractional step methods. A dive into the literature showed that much more powerful and promising multigrid algorithms were available, and the idea grew to address the full, nonlinear steady-state Navier-Stokes equations with complex features like adaptive grid refinement and, possibly, a solver for the eigenvalue problem associated with its linearization. In the end, that same idea had to be scaled back a little and the multigrid eigenvalue solver had to be dropped, but the Navier-Stokes solver has been successfully developed and applied to our problem.

Was the effort worth the result? There is no doubt I have learned a lot during this experience, both on the subject of multigrid, on how to organize my own work and possibly on how to do research. Then, the answer would look like a yes. But, as most things, what I have learned is relative to what I could have learned. There is a non-negligible possibility that, by investing less in code development (a tedious task indeed, in the end) and more in fluid dynamic analysis, which is the area of expertise of LadHyX, the whole process would have been more effective for everybody involved.

Probably, gone are the romantic days in which the lone cowboy (ehm, researcher) was writing his own code by punching cards. Writing good and, most important, reusable and well-documented code is now a huge task, and it is fairly sure someone else can do it better (the code!). Proofs abound: there are incredibly well-written and well-documented solvers and libraries for most computational tasks one can think of. Many of these solvers and libraries run on parallel computers with very little, if any, intervention. As a couple of examples, I would mention OpenFOAM as a well-established Navier-Stokes solver with plenty of models already implemented (turbulence, combustion, etc.) and to PETSc, SLEPc and Trilinos as general purpose libraries designed for scientific computation. And if you really "want to know what the code is doing" — the best excuse for writing your own code — you can just look at it, since all the mentioned libraries are open source. The investment in understanding how a well-documented (I repeat, well-documented) code is written is for sure less than what is required to write a code yourself and much

more useful from the learning point of view. If something needed is not there, it can be added, and it is always useful to have a structure to take inspiration from.

At this point I guess I should rewrite my code in a more ordered way and provide documentation.

But that is another story.

Financial support from
Airbus and CNRS/Ecole Polytechnique
is gratefully acknowledged

Luogo è là giù da Belzebù remoto
tanto quanto la tomba si distende,
che non per vista, ma per suono è noto

d'un ruscelletto che quivi discende
per la buca d'un sasso, ch'elli ha roso,
col corso ch'elli avvolge, e poco pende.

Lo duca e io per quel cammino ascoso
intrammo a ritornar nel chiaro mondo;
e sanza cura aver d'alcun riposo,

salimmo sù, el primo e io secondo,
tanto ch'i' vidi de le cose belle
che porta 'l ciel, per un pertugio tondo.

E quindi uscimmo a riveder le stelle.

Dante Alighieri — Inferno, Canto XXXIV

Appendix A

Matlab multigrid code

```
% A basic implementation of a multigrid
% solver for the convection-diffusion
% equation on a unit square.
%
% INPUT:
% - n: the number of mesh point along a mesh
% direction (the total number of degrees of
% freedom is n^2);
% - nits: the number of V-cycle to be used
%
% Both the CS and the FAS aglorithm are
% avaiable, and the used algorithm can be
% selected by setting the variable
% cntrparams.algorithm to 'CS' and 'FAS'
% respectively. The Poisson equation can be
% obtained by setting cntrparams.adv ---
% i.e. the coefficient of the convective
% term --- to zero. A purely convective
% equation can be obtained by setting
% cntrparam.nu --- i.e.  the viscosity ---
% to zero. Other input parameters are
% clarified in the code.
%
% OUTPUT:
% A single structure, named output, is
% provided as a result. Any quantity present
% in the main function can be easily added
% to the output structure.  The default
% quatities in the output structure are:
%
% - its: the number of V-cycle iteration
% [0:nits]
% - rnormRelaxation: the L2 norm of the
% residual for the relaxation scheme, as a
% function of the iteration number.
% (without multigrid)
% - rnormTwoGrids: the L2 norm of the
% residual for the two grid scheme, as a
% function of the iteration number.
% - rnormMG: the L2 norm of the residual for
% the multigrid scheme, as a function of the
% iteration number
% - x,y,uh,rh,u0: the x,y coordinate, the
% computed solution, the corresponding
% residual and the exact solution on the
% finest grid
% - Reh: the mesh-based Reynold number on
% the finest grid
%
% As a usage example, a plot of the
% convergence history of the residual for
% the multigrid scheme can be obtained with
%
% plot(output.its,output.rnormMG)
%


%
% the full code is made of the following
% functions:
%
% === [output] = main(n,nits) ===
% the main function
%
% === [J] = jac(n) ===
% returs the discretization of the
% convection-diffusion operator on a grid
% with n mesh points on each direction
%
% === [Lp,Lm] = relaxationSetUp(J,n) ===
% sets up the relaxation algorithm by
% performing the operator splitting and, if
% required, implementing the boundary
% relaxation
%
% === [U] = RR(u) ===
% restrict the field u to the next coarser
% grid
%
% === [u] = II(U) ===
% interpolate the field U to the next finer
% grid
%
% == [R] = Rop(n,N) ===
% provides the restriction operator between
% a finer and a corser grid characterized by
% n and N mesh points in each direction
% respectively.  Restriction can the be done
% as U = R*u and the result is the same as
```

```
% using U = RR(u). The implementation of RR
% is much faster, but this function can be
% used to obtain an explicit description of
% the restriction operator
%
% === [I] = Iop(N,n) ===
% provides the interpolation operator
% between a coarser and a finer grid
% characterized by N and n mesh points in
% each directio respectively.  The same
% observation made for Rop applies, with ... u =
% II(U) being a much faster implenentation
%
% === [uh] = mg(uh,rhs,n) ===
% this function represents the application
% of a single V-cycle. Both the CS and the
% FAS algorithm are implemented and can be
% selected by setting the value of
% cntrparams.algorithm to 'CS' or 'FAS'
% respectively.
%
% === [uh] = twogrids(uh,rhs,n) ===
% the two grid algorithm --- i.e. only two
% grids are used and the solution is
% computed on the coarses independently of
% its size. It can be used for comparison
% with mg.
%
% === [x,y,u0,rhs] = init(n) ===
% this function initialize the grid point
% coordinates, the exact solution and the
% corresponding right hand side. WARNING:
% some choices of boundary condition do not
% return a solution correspnding to the
% exact one.
%



function [output]= main(n,nits)

  %%%%%%%%%%%%%%%%%%%%%%%%%%%
  %%%% PROBLEM CONTROL %%%%%
  %%%%%%%%%%%%%%%%%%%%%%%%%%%

  global cntrparams
  % the algorithm used, can be 'CS' or 'FAS'
  cntrparams.algorithm = 'FAS';

  % the anisotropy parameter (see the
  % Relaxation section in the Multigrid
  % chapter). A value of epsilon < 1 is
  % correcly treated by the currect
  % implementation of linewise Gauss-Seidel
  % relaxation. A value of epsilon >1
  % correspond to stretching in the wrong
  % direction
  cntrparams.epsilon = 1;

  % control for the linewise Gauss-Seidel
  % relaxation. A value of 0 corresponds to
  % pointwise relaxation, a value of 1
  % correspond to linewise relaxation.
  cntrparams.LGS = 0;

  % the sweep direction of the relaxation,
  % can be 'F' (forward) or 'B' (backward).
  % A value of 'F' corresponds to a sweep
  % starting in the bottom left corner and
  % ending in the top right corner. A value
  % of 'B' correspond to a sweep in the
  % opposite direction, i.e. starting in the
  % top right corner and ending in the
  % bottom-left corner.
  cntrparams.sweepdirection = 'F';

  % the convective field, can be 'const' or
  % 'hiemenz'. The value 'const' correspond
  % to a constant flow field, the value
  % 'hiemenz' correspond to the (unswept)
  % hiemenz flow field.
  cntrparams.advfield = 'hiemenz';

  % the angle of the (constant) velocity
  % field with respect to the horizonal
  % axis.
  cntrparams.alph= 45*pi/180;

  % coefficient for the convection term. For
  % a value of zero the Poisson equation is
  % obtained. For a value of one, the
  % viscosity is the opposite of the
  % Reynolds number.
  cntrparams.adv = 0;

  % the order of discretization of the
  % advection term. Can be 1 or 2.
  cntrparams.advorder = 1;

  % the viscosity. It corresponds to the
  % inverse of the Reynolds number if
  % cntrparams.adv = 1. If one wants to
  % control the mesh-based Reynolds on the
  % finer grid, cntrparams.nu =
  % 1./(n-1)/Re_h, where Re_h is the
  % mesh-based Reynolds.
  cntrparams.nu = 1./(n-1)/1;

  % whether a Dirichlet (0) or a homogeneous
  % Neuamnn (1) boundary condition is
  % implemented on each boundary
```

```
cntrparams.neumannS = 0;
cntrparams.neumannN = 0;
cntrparams.neumannE = 0;
cntrparams.neumannW = 0;

% the number of mesh-lines to be included
% in the boundary relaxation on each
% boundary
cntrparams.brkS = 0;
cntrparams.brkN = 0;
cntrparams.brkW = 0;
cntrparams.brkE = 0;
cntrparams

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%% END OF PROBLEM CONTROL   %%%%%
%%%% YOU SHOULD NOT MODIFY    %%%%%
%%%% ANYTHING BELOW THIS LINE %%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


%%%%%%%%%%%%%%%%%%%%%%%%%
%%%% INITIALIZATION %%%%
%%%%%%%%%%%%%%%%%%%%%%%%%

% the total number of degrees of freeedom
nn = n*n;
% the discretized operator
Jh = jac(n);
% initialization of the rhs and exact
% solution
[x,y,u0,rhs] = init(n);

%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%% DIRECT SOLUTION %%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%
tic
disp('=== Direct solution ===')
u = u0;
u = Jh\rhs;
toc
output.u = reshape(u,n,n);

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%% SIMPLE RELAXATION %%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
tic
disp(['=== Simple relaxation ...
    (',num2str(3*nits),' iterations) ==='])
[Lp,Lm] = relaxationSetUp(Jh,n);
u = u0;
r = Jh*u-rhs;
rnormRelaxation(1) = norm(r)/nn;
for cc = 2:3*nits+1
  u = -Lp\(Lm*u-rhs);
  r = Jh*u-rhs;
  rnormRelaxation(cc) = norm(r)/nn;
end
toc

%%%%%%%%%%%%%%%%%%%%%%%%%
%%%% TWO GRID CYCLE %%%%
%%%%%%%%%%%%%%%%%%%%%%%%%
tic
disp(['=== Two grid cycle ...
    (',num2str(nits),' iterations) ==='])
uh = u0;
rh = Jh*uh - rhs;
rnormTwoGrids(1) = norm(rh)/nn;
for cc = 2:nits+1
  uh = twogrids(uh,rhs,n);
  rh = Jh*uh - rhs;
  rnormTwoGrids(cc) = norm(rh)/nn;
end
toc

%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%% MULTIGRID CYCLE %%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%
tic
disp(['=== Multigrid V-cycle ...
    (',num2str(nits),' iterations) ==='])
uh = u0;
rh = Jh*uh - rhs;
rnormMG(1) = norm(rh)/nn;
for cc = 2:nits+1
  uh = mg(uh,rhs,n);
  rh = Jh*uh - rhs;
  rnormMG(cc) = norm(rh)/nn;
  disp(['V cycle # ', num2str(cc), ' ...
      residual is ', ...
      num2str(rnormMG(cc))]);
end
toc

%%%%%%%%%%%%%%%%%
%%%% OUTPUT %%%%
%%%%%%%%%%%%%%%%%
output.its = 0:3*nits
output.rnormRelaxation = rnormRelaxation;
output.rnormTwoGrids = [rnormTwoGrids ...
    ones(1,nits*3-nits)*nan];
output.rnormMG = [rnormMG ...
    ones(1,nits*3-nits)*nan];
output.x = x;
output.y = y;
output.uh = reshape(uh,n,n);
output.rh = reshape(rh,n,n);
output.u0 = reshape(u0,n,n);
output.Reh = 1./(n-1)/cntrparams.nu;
dlmwrite('norms.dat', [output.its', ...
    output.rnormRelaxation', ...
```

```
      output.rnormTwoGrids', ...
      output.rnormMG'], 'delimiter',' ');

end

% function [J] = jac(n) returs the
% discretization of the convection-diffusion
% operator on a grid with n mesh points on
% each direction
function [J] = jac(n)
  global cntrparams

  epsilon = cntrparams.epsilon;
  alph = cntrparams.alph;
  nu = cntrparams.nu;
  adv = cntrparams.adv;
  advfield = cntrparams.advfield;
  advorder = cntrparams.advorder;
  neumannS = cntrparams.neumannS;
  neumannN = cntrparams.neumannN;
  neumannE = cntrparams.neumannE;
  neumannW = cntrparams.neumannW;

  nn = n*n;
  h = 1./(n-1);

  % vectors containing the index of the
  % boundary points
  sb = 1:n;
  nb = nn-n+1:nn;
  wb = 1:n:nn;
  eb = n:n:nn;
  bb = [ sb,nb,wb,eb];
  lbb = length(bb);

  % the centered second order discretization
  % of the Laplacian operator. epsilon
  % represents the anisotropy. spdiags will
  % be used to build the sparse matrix, look
  % at 'help spdiags' on Matlab for
  % information on the format used here.
  d0 = 1/h^2*[zeros(nn,1) , ...
      epsilon*ones(nn,1), zeros(nn,1),  ...
      ones(nn,1),  ...
      -2*(1+epsilon).*ones(nn,1),  ...
      ones(nn,1),zeros(nn,1),epsilon*ones(nn,1), ...
      zeros(nn,1)];

  % the convective the velocity field can be
  % constant, with an angle alph with the
  % horizontal axis, or (unswept) hiemenz.
  if (strcmpi(advfield,'const'))
    u = ones(nn,1).*cos(alph);
    v = ones(nn,1).*sin(alph);
  elseif (strcmpi(advfield,'hiemenz'))
    x = linspace(0,1,n); y = ...
      linspace(0,1,n); [x,y] = ...
      meshgrid(x,y);
    u = x; u = u(:);
    v = 1-y; v = v(:);
    clear x,y;
  else
    error('advfield can be const or ...
      hiemenz');
  end

  % the upwinded first or second order
  % discretization of the convective
  % operator
  up = .5*(u+abs(u)); um = -.5*(u-abs(u));
  vp = .5*(v+abs(v)); vm = -.5*(v-abs(v));
  d1 = zeros(nn,9);
  if (advorder==1)
    d1 = 1./(h)*[ 0*vp -1*vp 0*up -1*up ...
      (up+vp+um+vm) -1*um 0*um -1*vm ...
      0*vm ];
  elseif (advorder==2)
    d1 = 1./(2*h)*[ 1*vp -2*vp 1*up -2*up ...
      3*(up+vp+um+vm) -2*um 1*um -2*vm ...
      1*vm ];
  else
    error('The discretization order for ...
      the convective term must be 1 or 2');
  end

  % the discretized operator, including the
  % convective (d1) and diffusive (d0) terms
  % is built using spdiags
  J = spdiags(adv.*d1-nu*d0,[-2*n -n -2 ...
     -1 0 1 2 n 2*n],nn,nn);

  % enforce Dirichlet boundary conditions on
  % all boundaries. The indexes of the
  % boundary's points are contained in the
  % vector bb
  J(bb,:) = 0;
  J(bb,bb) = spdiags(ones(lbb,1),0,lbb,lbb);

  % enforce Neumann boundary condition where
  % required
  if (neumannS == 1)
... J(sb,:) = 0;
    for i = 1:n
      J(sb(i),sb(i)+[0 n 2*n]) = -1/(2*h) ...
        * [ 3 -2 1];
    end
  end

  if (neumannE == 1)
    J(eb,:) = 0;
    for i = 1:n
      J(eb(i),eb(i)-[0 1 2]) = +1/(2*h) * ...
```

```
            [ 3 -2 1];
    end
  end

  if (neumannN == 1)
    J(nb,:) = 0;
    for i = 1:n
      J(nb(i),nb(i)-[0 n 2*n]) = +1/(2*h) ...
          * [ 3 -2 1];
    end
  end

  if (neumannW == 1)
    J(wb,:) = 0;
    for i = 1:n
      J(wb(i),wb(i)+[0 1 2]) = -1/(2*h) * ...
          [ 3 -2 1];
    end
  end

end

% function [Lp,Lm] = relaxationSetUp(J,n);
% sets up the relaxation algorithm by
% performing the operator splitting and, if
% required, implementing the boundary
% relaxation
function [Lp,Lm] = relaxationSetUp(J,n);
  global cntrparams;

  LGS = cntrparams.LGS;

  % perform operator splitting by selecting
  % the upper and lower parts of the matrix.
  % LGS diagonals above/below the main
  % diagonal are included in the Lp operator
  % in order to implenent linewise
  % Gauss-Seidel, if required.
  if (cntrparams.sweepdirection == 'F')
    Lp = tril(J,+LGS);
    Lm = triu(J,+1+LGS);
  elseif (cntrparams.sweepdirection == 'B')
    Lp = triu(J,-LGS);
    Lm = tril(J,-1-LGS);
  end

  % boundary relaxation is obtained
  % modifying the splitting. All discrete
  % equations corredponding to the boundary
  % points are included in the Lp operator
  % and are solved implicitly.

  % southern boundary
  brkS = cntrparams.brkS;
  Lp(1:brkS*n,:) = J(1:brkS*n,:);
  Lm(1:brkS*n,:) = 0;

  % northern boundary
  brkN = cntrparams.brkN;
  Lp(end-brkN*n+1:end,:) = ...
      J(end-brkN*n+1:end,:);
  Lm(end-brkN*n+1:end,:) = 0;

  % western boundary
  brkW = cntrparams.brkW;
  brkpoints = zeros(brkW*n,1);
  for i = 1:brkW
    brkpoints((i-1)*n+1:i*n) = i:n:nn;
  end
  Lp(brkpoints,:) = J(brkpoints,:);
  Lm(brkpoints,:) = 0;

  % eastern boundary
  brkE = cntrparams.brkE;
  brkpoints = zeros(brkE*n,1);
  for i = 1:brkE
    brkpoints((i-1)*n+1:i*n) = n-i+1:n:nn;
  end
  Lp(brkpoints,:) = J(brkpoints,:);
  Lm(brkpoints,:) = 0;

end

% function U = RR(u)
% restrict the field u to the next coarser
% grid
function U = RR(u)
  n = sqrt(length(u));
  N = ceil(n/2);
  utmp = reshape(u,n,n);
  Utmp = zeros(N,N);

  Utmp(2:N-1,2:N-1) = ...
      0.25*utmp(3:2:n-2,3:2:n-2) + ...
    0.125*( utmp(2:2:n-3,3:2:n-2) + ...
        utmp(4:2:n-1,3:2:n-2) ...
    + utmp(3:2:n-2,2:2:n-3) + ...
        utmp(3:2:n-2,4:2:n-1) ) + ...
    0.0625*( utmp(2:2:n-3,2:2:n-3) + ...
        utmp(2:2:n-3,4:2:n-1) ...
    + utmp(4:2:n-1,2:2:n-3) + ...
        utmp(4:2:n-1,4:2:n-1) );

  Utmp(2:N-1,1) = 0.5*utmp(3:2:n-2,1) + ...
      0.25*( utmp(2:2:n-3,1) + ...
      utmp(4:2:n-1,1) );
  Utmp(2:N-1,N) = 0.5*utmp(3:2:n-1,n) + ...
      0.25*( utmp(2:2:n-3,n) + ...
      utmp(4:2:n-1,n) );
  Utmp(1,2:N-1) = 0.5*utmp(1,3:2:n-1) + ...
      0.25*( utmp(1,2:2:n-3) + ...
      utmp(1,4:2:n-1) );
```

```
  Utmp(N,2:N-1) = 0.5*utmp(n,3:2:n-1) + ...
      0.25*( utmp(n,2:2:n-3) + ...
      utmp(n,4:2:n-1) );

  Utmp(1,1) = utmp(1,1);
  Utmp(1,end) = utmp(1,end);
  Utmp(end,1) = utmp(end,1);
  Utmp(end,end) = utmp(end,end);

  U = reshape(Utmp,N*N,1);

end

% function u = II(U)
% interpolate the field U to the next finer
% grid
function u = II(U)
  N = sqrt(length(U));
  n = 2*N-1;
  Utmp = reshape(U,N,N);
  utmp = zeros(n,n);

  utmp(1:2:n,1:2:n) = Utmp(1:N,1:N);
  utmp(2:2:n-1,1:2:n) = ( Utmp(1:N-1,1:N) ...
      + Utmp(2:N,1:N) ) / 2.;
  utmp(1:2:n,2:2:n-1) = ( Utmp(1:N,1:N-1) ...
      + Utmp(1:N,2:N) ) / 2.;
  utmp(2:2:n-1,2:2:n-1) = ( ...
      Utmp(1:N-1,1:N-1) + Utmp(1:N-1,2:N) ...
      + Utmp(2:N,1:N-1) + Utmp(2:N,2:N) ) ...
      / 4.;
  u = reshape(utmp,n*n,1);

end


% function [R] = Rop(n,N)
% provides the restriction operator between
% a finer and a corser grid characterized by
% n and N mesh points in each direction
% respectively.  Restriction can the be done
% as U = R*u and the result is the same as
% using U = RR(u). The implementation of RR
% is much faster, but this function can be
% used to obtain an explicit description of
% the restriction operator
function [R] = Rop(n,N)
  nn = n*n;
  NN = N*N;

  % SW S SE W C E NW N E
  st = [-n-1 -n -n+1  -1 0 1 n-1 n n+1];

  R = sparse([],[],[],NN,nn,nn*9);
  di = 1/16*[ 1 , 2 , 1. , 2 , 4 , 2 , 1 ...
      , 2 , 1] ; % interior
  dh = 1/4*[ 0 , 0 , 0. , 1 , 2 , 1 , 0 , ...
      0 , 0] ; % horizontal boundaries
  dv = 1/4*[ 0 , 1 , 0. , 0 , 2 , 0 , 0 , ...
      1 , 0] ; % vertical boundaries
  dc = [ 0 , 0 , 0. , 0 , 1 , 0 , 0 , 0 , ...
      0] ; % corners

  for J = 2:N-1
    for I = 2:N-1
      j = 2*J-1;
      i = 2*I-1;
      CC = (J-1)*N+I;
      cc = (j-1)*n+i;
      R(CC,st+cc)  = di;
    end
  end

  for J = 1:N-1:N
    for I = 2:N-1
      j = 2*J-1;
      i = 2*I-1;
      CC = (J-1)*N+I;
      cc = (j-1)*n+i;
      R(CC,:) = spdiags(dh,st+cc-1,1,nn);
    end
  end

  for J = 2:N-1
    for I = 1:N-1:N
      j = 2*J-1;
      i = 2*I-1;
      CC = (J-1)*N+I;
      cc = (j-1)*n+i;
      R(CC,:) = spdiags(dv,st+cc-1,1,nn);
    end
  end

  for J = 1:N-1:N
    for I = 1:N-1:N
      j = 2*J-1;
      i = 2*I-1;
      CC = (J-1)*N+I;
      cc = (j-1)*n+i;
      R(CC,:) = spdiags(dc,st+cc-1,1,nn);
    end
  end

end


% function [Iout] = Iop(N,n)
% provides the interpolation operator
% between a coarser and a finer grid
% characterized by N and n mesh points in
% each directio respectively.  The same
% observation made for Rop applies, with ...
```

```
    u =
% II(U) being a much faster implenentation
function [Iout] = Iop(N,n)
  nn = n*n;
  NN = N*N;

  % SW S SE W C E NW N E
  st = [-n-1 -n -n+1  -1 0 1 n-1 n n+1];

  Iout = sparse([],[],[],nn,NN,nn.*9);
  di = 1/4*[ 1 , 2 , 1. , 2 , 4 , 2 , 1 , ...
      2 , 1] ; % interior
  dh = 1/2*[ 0 , 0 , 0. , 1 , 2 , 1 , 0 , ...
      0 , 0] ; % horizontal boundaries
  dv = 1/2*[ 0 , 1 , 0. , 0 , 2 , 0 , 0 , ...
      1 , 0] ; % vertical boundaries
  dc = [ 0 , 0 , 0. , 0 , 1 , 0 , 0 , 0 , ...
      0] ; % corners

  % injection
  for J = 1:N
    for I = 1:N
      j = 2*J-1;
      i = 2*I-1;
      CC = (J-1)*N+I;
      cc = (j-1)*n+i;
      Iout(cc,CC) = 1.;
    end
  end

  % linear interpolation
  for J = 1:N
    for I = 1:N-1
      j = 2*J-1;
      i = 2*I;
      CC = (J-1)*N+I;
      cc = (j-1)*n+i;
      Iout(cc,CC) = 0.5;
      Iout(cc,CC+1) = 0.5;
    end
  end

  for J = 1:N-1
    for I = 1:N
      j = 2*J;
      i = 2*I-1;
      CC = (J-1)*N+I;
      cc = (j-1)*n+i;
      Iout(cc,CC) = 0.5;
      Iout(cc,CC+N) = 0.5;
    end
  end

  % bilinear interpolation
  for J = 1:N-1
    for I = 1:N-1
      j = 2*J;
      i = 2*I;
      CC = (J-1)*N+I;
      cc = (j-1)*n+i;
      Iout(cc,CC) = 0.25;
      Iout(cc,CC+1) = 0.25;
      Iout(cc,CC+N) = 0.25;
      Iout(cc,CC+N+1) = 0.25;
    end
  end
end


function [uh] = mg(uh,rhs,n)

  global cntrparams
  nn = n*n;
  LGS = cntrparams.LGS;
  if (n <= 5)

    Jh = jac(n);
    uh = Jh\rhs;

  else

    N = (n-1)/2+1;
    % the follwing two lines can be
    % uncommented if an explicit
    % representation of the restriction and
    % iterpolation operator are required,
    % but the implentation of the Rop and
    % Iop functions is quite slow
    %R = Rop(n,N);
    %I = Iop(N,n);
    Jh = jac(n);
    JH = jac(N);

    [Lp,Lm] = relaxationSetUp(Jh,n);

    % relaxation: relaxation is peformed by
    % "inverting" the Lp operator. This is
    % more costly than a real Gauss-Seidel
    % iteration, but clearer from a notation
    % point of view and, for this reason, is
    % used here.
    uh = -Lp\(Lm*uh-rhs);
    uh = -Lp\(Lm*uh-rhs);

    if ( ...
        strcmpi(cntrparams.algorithm,'FAS') ...
        )  % the FAS algorithm

      % restriction
      tau = JH*(RR(uh)) - RR(Jh*uh);
      rhsH = RR(rhs) + tau;

```

```
      % multigrid on coarser grids
      uHold = RR(uh);
      uH = mg(uHold,rhsH,N);

      % interpolation
      corrH = uH - uHold;
      uh = uh + II(corrH);

    elseif ( ...
        strcmpi(cntrparams.algorithm,'CS') ...
        ) % the CS algorithm

      rh = rhs - Jh*uh;
      rhsH = RR(rh);
      corrH = mg(zeros(size(rhsH)),rhsH,N);
      uh = uh + II(corrH);

    else
      error('cntrparams.algorithm must be ...
          FAS or CS');
    end

    % relaxation
    uh = -Lp\(Lm*uh-rhs);

  end

end

function [uh] = twogrids(uh,rhs,n)

  global cntrparams;
  nn = n*n;
  LGS = cntrparams.LGS;

  N = (n-1)/2+1;

  %R = Rop(n,N);
  %I = Iop(N,n);
  Jh = jac(n);
  JH = jac(N);

  [Lp,Lm] = relaxationSetUp(Jh,n);

  % relaxation
  uh = -Lp\(Lm*uh-rhs);
  uh = -Lp\(Lm*uh-rhs);

  if ( ...
      strcmpi(cntrparams.algorithm,'FAS') ...
      )  % the FAS algorithm

    % restriction
    tau = JH*RR(uh) - RR(Jh*uh);
    rhsH = RR(rhs) + tau;

    % solve on coarse grid
    uH = JH\rhsH;

    % interpolation
    corrH = uH - RR(uh);
    uh = uh + II(corrH);

  elseif ( ...
      strcmpi(cntrparams.algorithm,'CS') ...
      ) % the CS algorithm

    rh = rhs - Jh*uh;
    corrH = JH\(RR(rh));
    uh = uh + II(corrH);

  else
    error('cntrparams.algorithm must be ...
        FAS or CS');
  end

  % relaxation
  uh = -Lp\(Lm*uh-rhs);

end

function [x,y,u0,rhs] = init(n)
  global cntrparams

  x = linspace(0,1,n); y = ...
      linspace(0,1,n); [x,y] = meshgrid(x,y);
  nu = cntrparams.nu; adv = ...
      cntrparams.adv; alph = cntrparams.alph;

  u0 = (x.^2 - x.^4) .* (y.^4-y.^2);   ...
      u0(2:end-1,2:end-1) = ...
      rand(n-2,n-2); u0 = u0(:);
  %u0 = (x.^2 - x.^4) .* (y.^4-y.^2);    ...
      u0 = u0(:);

  rhs = + adv.*( ...
      cos(alph).*(2.*x-4.*x.^3).*(y.^4-y.^2) ...
      + ...
      sin(alph).*(x.^2-x.^4).*(4.*y.^3-2.*y) ...
      )  - nu.*( ...
      (2-12.*x.^2).*(y.^4-y.^2) + ...
      (x.^2-x.^4).*(12.*y.^2-2) );
  rhs(:,1) = 0; rhs(:,end)=0; rhs(1,:) = ...
      0; rhs(end,:) = 0; rhs = rhs(:);
end
```

Bibliography

[1] P. R. Amestoy, I. S. Duff, J. Koster, and J.-Y. L'Excellent. A fully asynchronous multifrontal solver using distributed dynamic scheduling. *SIAM Journal on Matrix Analysis and Applications*, 23(1):15–41, 2001.

[2] P. R. Amestoy, A. Guermouche, J.-Y. L'Excellent, and S. Pralet. Hybrid scheduling for the parallel solution of linear systems. *Parallel Computing*, 32(2):136–156, 2006.

[3] E. Anderson, Z. Bai, C. Bischof, S. Blackford, J. Demmel, J. Dongarra, J. Du Croz, A. Greenbaum, S. Hammarling, A. McKenney, and D. Sorensen. *LAPACK Users' Guide*. Society for Industrial and Applied Mathematics, Philadelphia, PA, third edition, 1999.

[4] Satish Balay, Jed Brown, , Kris Buschelman, Victor Eijkhout, William D. Gropp, Dinesh Kaushik, Matthew G. Knepley, Lois Curfman McInnes, Barry F. Smith, and Hong Zhang. PETSc users manual. Technical Report ANL-95/11 - Revision 3.3, Argonne National Laboratory, 2012.

[5] Satish Balay, Jed Brown, Kris Buschelman, William D. Gropp, Dinesh Kaushik, Matthew G. Knepley, Lois Curfman McInnes, Barry F. Smith, and Hong Zhang. PETSc Web page, 2012. http://www.mcs.anl.gov/petsc.

[6] Satish Balay, William D. Gropp, Lois Curfman McInnes, and Barry F. Smith. Efficient management of parallelism in object oriented numerical software libraries. In E. Arge, A. M. Bruaset, and H. P. Langtangen, editors, *Modern Software Tools in Scientific Computing*, pages 163–202. Birkhäuser Press, 1997.

[7] A. Brandt. Multi-level adaptive solutions to boundary-value problems. *Mathematics of Computation*, 31(138):333–390, 1977.

[8] A. Brandt. *Multigrid techniques: 1984 guide with applications to fluid dynamics*. GMD Bonn, 1984.

[9] A. Brandt. Multiscale scientific computation: Review 2001. *Multiscale and Multiresolution Methods: Theory and Applications*, page 3, 2002.

[10] A. Brandt. Multiscale calculation of many eigenfuctions. Technical report, Weizmann institute of science, 2003.

[11] A. Brandt, S. McCormick, and J. Ruge. Multi-grid methods for differential eigenproblems. *SIAM J. Sci. Statist. Comput.*, 4:244–260, 1983.

[12] J.M. Chomaz. Global instabilities in spatially developing flows: non-normality and nonlinearity. *Annu. Rev. Fluid Mech.*, 37:357–392, 2005.

[13] J.R. Dagenhart and W.S. Saric. Crossflow stability and transition experiments in swept-wing flow. Technical report, NASA Langley Research Center, Hampton, Virginia, 1999.

[14] B. Diskin, J.L. Thomas, and R.E. Mineck. Textbook multigrid efficiency for leading edge stagnation. Technical report, NASA Langley Research Center, Hampton, Virginia, 2004.

[15] P.G. Drazin and N. Riley. *The Navier-Stokes equations: a classification of flows and exact solutions*, volume 334. Cambridge University Press, 2006.

[16] B. Fornberg. A numerical study of steady viscous flow past a circular cylinder. *Journal of Fluid Mechanics*, 98(04):819–855, 1980.

[17] M. Gaster. A simple device for preventing turbulent contamination on swept leading edges. *Journal of the Royal Aeronautical Society*, 69:788, 1965.

[18] F. Giannetti and P. Luchini. Receptivity of the circular cylinder's first instability. In *Proc. 5th Eur. Fluid Mech. Conf., Toulouse*, 2003.

[19] F. Giannetti and P. Luchini. Structural sensitivity of the first instability of the cylinder wake. *Journal of Fluid Mechanics*, 581(1):167–197, 2007.

[20] P.M. Gresho. Incompressible fluid dynamics: some foundamental formulation issues. *Annual Review of Fluid Mechanics*, 23(1):413–453, 1991.

[21] P.M. Gresho and R.L. Sani. On pressure boundary conditions for the incompressible navier-stokes equations. *International Journal for Numerical Methods in Fluids*, 7(10):1111–1145, 1987.

[22] A. Guégan, P. Huerre, and P.J. Schmid. Optimal disturbances in swept hiemenz flow. *J Fluid Mechanics*, 578:223–232, 2007.

[23] A. Guégan, P.J. Schmid, and P. Huerre. Optimal energy growth and optimal control in swept hiemenz flow. *Journal of Fluid Mechanics*, 566(1):11–45, 2006.

[24] Alan Guégan. *Optimal perturbations in swept leading-edge boundary layers*. PhD thesis, École polytechnique Laboratoire d'hydrodynamique (LadHyX), 2007.

[25] P. Hall, M.R. Malik, and Dia Poll. On the stability of an infinite swept attachment line boundary layer. *Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences*, 395(1809):229–245, 1984.

[26] P. Hall and S.O. Seddougui. Wave interactions in a three-dimensional attachment-line boundary layer. *Journal of Fluid Mechanics*, 217:367–390, 1990.

[27] V. Hernandez, J. E. Roman, and V. Vidal. SLEPc: Scalable Library for Eigenvalue Problem Computations. *Lecture Notes in Computer Science*, 2565:377–391, 2003.

[28] V. Hernández, J.E. Román, A. Tomás, and V. Vidal. Slepc users manual. Technical report, Technical Report DSIC-II/24/02-Revision 2.3. 2, D. Sistemas Informáticos y Computación, Universidad Politécnica de Valencia, 2006.

[29] V. Hernández, J.E. Román, A. Tomás, and V. Vidal. SLEPc Web page, 2012. http://www.grycap.upv.es/slepc/.

[30] Vicente Hernandez, Jose E. Roman, and Vicente Vidal. SLEPc: A scalable and flexible toolkit for the solution of eigenvalue problems. *ACM Transactions on Mathematical Software*, 31(3):351–362, 2005.

[31] K. Hiemenz. Boundary layer for a homogeneous flow around a dropping cylinder. *Dinglers Polytechnic Journal*, 326:321–324, 1911.

[32] R.D. Joslin. Direct simulation of evolution and control of three-dimensional instabilities in attachment-line boundary layers. *Journal of Fluid Mechanics*, 291:369–392, 1995.

[33] R.D. Joslin. Simulation of three-dimensional symmetric and asymmetric instabilitiens in attachment-line boundary layers. *AIAA Journal*, 34(11):2432–2434, 1996.

[34] R.D. Joslin. Aircraft laminar flow control. *Annual review of fluid mechanics*, 30(1):1–29, 1998.

[35] R.D. Joslin. Overview of laminar flow control. Technical report, NASA Langley Research Center, Hampton, Virginia, 1998.

[36] R. Kettler. Analysis and Comparison of Relaxation Schemes in Robust Multigrid and Preconditioned Conjugate Gradient Methods. In *Multigrid methods*, volume 960, pages 502–534. Springer, 1982.

[37] D. Kushnir, M. Galun, and A. Brandt. Efficient multilevel eigensolvers with applications to data analysis tasks. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 32(8):1377–1391, 2010.

[38] R.S. Lin and M.R. Malik. On the stability of attachment-line boundary layers. part 1. the incompressible swept hiemenz flow. *Journal of Fluid Mechanics*, 311:239–256, 1996.

[39] R.S. Lin and M.R. Malik. On the stability of attachment-line boundary layers. part 2. the effect of leading-edge curvature. *Journal of Fluid Mechanics*, 333:125–137, 1997.

[40] C.J. Mack. *Global stability of compressible flow about a swept parabolic body.* PhD thesis, École polytechnique Laboratoire d'hydrodynamique (LadHyX), 2009.

[41] C.J. Mack and P.J. Schmid. Direct numerical study of hypersonic flow about a swept parabolic body. *Computers & Fluids*, 39(10):1932–1943, 2010.

[42] C.J. Mack and P.J. Schmid. Global stability of swept flow around a parabolic body: features of the global spectrum. *Journal of Fluid Mechanics*, 669:375–396, 2011.

[43] C.J. Mack and P.J. Schmid. Global stability of swept flow around a parabolic body: the neutral curve. *Journal of Fluid Mechanics*, 678:589–599, 2011.

[44] C.J. Mack, P.J. Schmid, and J.L. Sesterhenn. Global stability of swept flow around a parabolic body: connecting attachment-line and crossflow modes. *Journal of Fluid Mechanics*, 611:205–214, 2008.

[45] O Marquet, D Sipp, and L Jacquin. Sensitivity analysis and passive control of cylinder flow. *Journal of Fluid Mechanics*, 615:221–252, 2008.

[46] M. McIntyre. Lucidity and science i: Writing skills and the pattern perception hypothesis. *Interdisciplinary science reviews*, 22(3):199–216, 1997.

[47] M.V. Morkovin. On the many faces of transition. *Viscous Drag Reduction*, pages 1–31, 1969.

[48] D. Obrist and P.J. Schmid. On the linear stability of swept attachment-line boundary layer flow. part 1. spectrum and asymptotic behaviour. *Journal of Fluid Mechanics*, 493:1–29, 2003.

[49] D. Obrist and P.J. Schmid. On the linear stability of swept attachment-line boundary layer flow. part 2. non-modal effects and receptivity. *Journal of Fluid Mechanics*, 493:31–58, 2003.

[50] D. Obrist and P.J. Schmid. Algebraically decaying modes and wave packet pseudo-modes in swept hiemenz flow. *Journal of Fluid Mechanics*, 643(1):309–332, 2010.

[51] W. Pfenninger. Flow phenomena at the leading edge of swept wings. *Recent Developments in Boundary Layer Research, AGARDograph*, 97(4), 1965.

[52] O. Pironneau, F. Hecht, AL Hyaric, and K. Ohtsuka. Freefem, 2005.

[53] H.L. Reed and W.S. Saric. Stability of three-dimensional boundary layers. *Annual Review of Fluid Mechanics*, 21(1):235–284, 1989.

[54] H.L. Reed, W.S. Saric, and D. Arnal. Linear stability theory applied to boundary layers. *Annual review of fluid mechanics*, 28(1):389–428, 1996.

[55] CW Rowley. Model Reduction for fluids, using balanced proper orthogonal decomposition. *To appear in Int. J. on Bifurcation and Chaos*, 2005.

[56] RL Sani, J. Shen, O. Pironneau, and PM Gresho. Pressure boundary condition for the time-dependent incompressible navier–stokes equations. *International Journal for Numerical Methods in Fluids*, 50(6):673–682, 2006.

[57] W.S. Saric, H.L. Reed, and E.J. Kerschen. Boundary-layer receptivity to freestream disturbances. *Annual review of fluid mechanics*, 34(1):291–319, 2002.

[58] W.S. Saric, H.L. Reed, and E.B. White. Stability and transition of three-dimensional boundary layers. *Annual Review of Fluid Mechanics*, 35(1):413–440, 2003.

[59] P.J. Schmid. Nonmodal Stability Theory. *Annual Review of Fluid Mechanics*, 39:129, 2007.

[60] P.J. Schmid. Dynamic mode decomposition of numerical and experimental data. *Journal of Fluid Mechanics*, 656(1):5–28, 2010.

[61] P.J. Schmid and D.S. Henningson. *Stability and transition in shear flows*, volume 142. Springer Verlag, 2001.

[62] David Sidilkover and Uri M. Ascher. A multigrid solver for the steady state navier-stokes equations using the pressure-poisson formulation. *Comp. Appl. Math.*, 1995.

[63] Herbert L. Stone. Iterative solution of implicit approximations of multidimensional partial differential equations. *SIAM Journal on Numerical Analysis*, 5(3):530–558, 1968.

[64] R.C. Swanson. Evaluation of a multigrid scheme for the incompressible navier–stokes equations. Technical report, NASA Langley Research Center, Hampton, Virginia, 2004.

[65] J.L. Thomas, B. Diskin, and A. Brandt. Textbook multigrid efficiency for fluid simulations. *Annual review of fluid mechanics*, 35(1):317–340, 2003.

[66] U. Trottenberg, C.W. Oosterlee, and A. Schüller. *Multigrid*. Academic Pr, 2001.