UNIVERSITÄT SIEGEN

Daniel Marinc

Analysis of adjoint based optimal control applied to noise reduction of plane jets.

Herausgeber: Holger Foysi

Schriftenreihe des Lehrstuhls für Strömungsmechanik

Band I

Analysis of adjoint based optimal control applied to noise reduction of plane jets.

Dem Department Maschinenbau
der Universität Siegen
zur Erlangung des akademischen Grades

Doktor-Ingenieur (Dr.-Ing.)

vorgelegte Dissertation
von

Dipl.-Phys. Daniel Marinc

Hauptreferent:	Prof. Dr.-Ing. H. Foysi
Koreferenten:	Prof. Dr.-Ing. W. Schröder
	Prof. Dr.-Ing. C. Fritzen
Vorsitzender:	Prof. Dr.-Ing. T. Seeger
Dekan:	Prof. Dr. rer. nat. U. Pietsch

Tag der Einreichung: 12.02.2014
Tag der mündl. Prüfung: 24.06.2014

Siegen, September 2014

gedruckt auf alterungsbeständigem holz- und säurefreiem Papier

Impressum
Prof. Dr.-Ing., Dipl.-Phys. Holger Foysi
Lehrstuhl für Strömungsmechanik
Universität Siegen
57068 Siegen
Band 1
Zugl.: Siegen, Univ., Diss. 2014

"Einfach reden, aber kompliziert denken - Nicht umgekehrt."

Franz Josef Strauß

"I know that self-promotion happens a lot and if people want to do that, good luck to them, but I do not regard it as a positive thing."

Grigorij Perelman

"Genius is one per cent inspiration, ninety-nine per cent perspiration"

Thomas Edison

Abstract

Many gradient based optimization methods can be found in the literature. If the input parameters of a cost function are subject to implicit constraints the adjoint method is often the preferred method to obtain the gradient. The adjoint method provides an efficient way for the computation of the gradient, where the computational time is independent of the number of control variables of the cost function.

In this work the noise of a three dimensional turbulent subsonic jet is minimized. The cost function is defined as an integral over the pressure fluctuations in the farfield and thus gives an estimate for noise emission. The simulations are based on the compressible Navier-Stokes equations, which act as implicit constraints for the minimization problem. The adjoint variables are computed using different approximations of the adjoint equations and the resulting gradient accuracy is investigated. Among other aspect the continuous and discrete adjoint methods are compared in terms of accuracy. Furthermore, the efficiency and effectivity of the optimization procedure is investigated in terms of gradient accuracy and numeric resolution.

Keywords: Aeroacoustics, Adjoint Method, Flow Control, Noise Minimization, Continuous Adjoint, Discrete Adjoint

Zusammenfassung

In der Literatur kann eine Vielzahl an Methoden zur gradientenbasierten numerischen Optimierung gefunden werden. Für den Fall, dass die Parameter einer Kostenfunktion impliziten Bedingungen unterliegen, wird häufig die Adjungiertenmethode zur Berechnung des Gradienten herangezogen. Die Adjungiertenmethode erlaubt auch bei impliziten Bedingungen eine effiziente Berechnung des Gradienten, wobei die benötigte Rechenzeit unabhängig von der Anzahl der Parameter der Kostenfunktion ist.

Im Rahmen dieser Arbeit wird der Schall eines dreidimensionalen turbulenten unterschall Freistrahles minimiert. Die Kostenfunktion wird definiert als ein Integral über die Druckfluktuationen im Fernfeld und ist somit ein Maß für die Lautstärke des Freistrahles. Die zugrundeliegenden Bewegungsgleichungen sind die kompressiblen Navier-Stokes-Gleichungen, welche implizite Bedingungen aus Sicht des Optimierungsprozesses sind. Die Adjungierte wird auf unterschiedliche Arten berechnet und die Genauigkeit des daraus resultierenden Gradienten untersucht werden. Ein besonderes Augenmerk liegt hierbei auf Vergleichen zwischen der sogenannten diskreten und kontinuierlichen Adjungierten. Desweiteren wird die Effizienz und Effektivität der Optimierung in Abhängigkeit der Genauigkeit des Gradienten und der numerischen Auflösung betrachtet.

Schlagwörter: Aeroakustik, Adjungierten Methode, Strömungskontrolle, Schallreduktion, Kontinuierliche Adjungierte, Diskrete Adjungierte

Contents

Nomenclature

$\mathbf{\Phi}$	vector of state variables
$\mathbf{R}$	right hand side of equations of motion
$\cdot'$	variation with respect to a control variation g'
$\cdot^{\circ}$	variation with respect to a control variation g°
γ	ratio of specific heats
$\Im$	cost functional
λ	heat conductivity
$\mathbf{g}$	vector of control variables
$\mathbf{U}$	vector containing the flow variables density, velocities and pressure
μ	dynamic viscosity
$\bar{\cdot}$	temporal average (sometimes also spatial average, see context)
ρ	mass density
τ_{ij}	component of shear stress tensor
Θ	Heaviside step function
$\widehat{\cdot}$	Reynolds fluctuations
c	speed of sound
C_p	specific heat capacity at constant pressure

C_v	specific heat capacity at constant volume
D	jet diameter in inflow plane
L	Lagrangian
m_i	momentum density in direction i
Ma	Mach number
p	static pressure
Pr	Prandtl number
R	ideal gas constant
Re	Reynolds number
T	temperature or time interval length
T_{suth}	Sutherland temperature
u_i	velocity in direction i
U_j	jet velocity in inflow plane
x_i	coordinate in direction i

Abbreviations

POD	proper orthogonal decomposition
BPOD	balanced proper orthogonal decomposition
DMD	dynamic mode decomposition
ROM	reduced order model
LQG	linear model, quadratic cost function, gaussian noise
UQ	uncertainty quantification
BC	boundary conditions
CBC	characteristic boundary conditions
NBC	non-reflecting boundary conditions
RK	Runge-Kutta
FD	finite differences
LHS	left hand side
RHS	right hand side
PDE	partial differential equation
LBFGS	low storage BFGS
CG	conjugate gradient
CGLin	conjugate gradient for linear systems

BFGS	Broyden-Fletcher-Goldfarb-Shanno
NCGtrust	Newton conjugate gradient with trust region
NCGline	Newton conjugate gradient with line search
NLan	Newton Lanczos
DNS	direct numerical simulation
LES	large eddy simulation
RANS	Reynolds averaged Navier Stokes
DRP	dispersion relation preserving
AD	automatic differentiation
CFL	Courant-Friedrichs-Lewy
SPL	sound pressure level
I/O	input/output
SAT	simultaneous approximation term
SBP	summation by parts

1. Introduction

1.1. Motivation

Flow control is an area with many fields of practical applications. Examples cover many different areas of aerodynamics such as shape optimization [54, 1, 16, 19, 71, 85, 107, 68, 61], boundary control [5, 22, 23, 24, 63, 64] or noise control [103, 84, 2, 50, 77, 78, 59]. In control theory one can distinguish between closed and open loop control. In the latter case one aimes to find an optimal control. In closed loop control one seeks an optimal feedback rule between some measurement of the state space and the control.

Another categorization of control problems is based on the method used to identify effective control parameters. Many different approaches have been investigated to influence flows in a desired way. Parameter studies, often in conjunction with physical reasoning, play an important role in optimization as its approach is quite simple [26, 106, 79, 9]. However, parameter studies are usually time intensive and due to the complex nature of the Navier-Stokes equations simple models based on physical reasoning are often not available. An improvement over simple parameter variations are derivative free optimization methods, e.g. surrogate management framework [61], simplex methods [74] or evolutionary strategies [34]. Instead of controlling the complete flow system at hand a reduced order model (ROM) can be derived. Due to the decreased state space dimension this ROMs are usually easier to control. This method relies strongly on the capability of finding a basis small compared to the state space dimension, but nevertheless capable of reflecting the relevant physical effects of the original system. A prominent choice for such a basis are proper orthogonal decomposition (POD) modes [4, 76, 55]. Several extensions of the POD approach exist to make the resulting ROM more robust regarding parameter variations, e.g. balanced proper orthogonal decomposition (BPOD) modes [3, 41], Taylor series expansion [41] or stochastic parameter variation [63]. Another choice for a ROM basis could be based on dynamic mode decomposition (DMD) [81]. An interesting application for ROMs is the linear model, quadratic cost function, gaussian noise (LQG) framework [24, 3]. LQG leads to a Riccati equation, whose solution gives an optimal feedback estimate based on a given flow measure. However, the linearization of the model equations restricts the area of applicability of this approach. In uncertainty quantification (UQ) methods, such as polynomial chaos [52, 35, 67] or others [28], instead of assigning a fixed value to all parameters some parameters are left uncertain. Usually this uncertainty is interpreted as stochastic noise of unknown or unquantifiable magnitude. However, UQ is capable of delivering the response of a cost functional with respect to the variation of some parameters. Thus, UQ methods might also be used to identify an appropriate control strategy.

Another control method is to exploit information about the derivatives of a given cost functional for its optimization. Derivative based optimization is a topic well documented in literature. Several classes of efficient and robust schemes are known. However, the computation of derivatives can be implementationally challenging and computationally time consuming. This holds especially for systems subject to conservation laws, as these conservation laws usually constitute implicit constraints for the control parameters. In the presence of implicit constraints derivatives can be computed by solving the linearized state equations resulting from the conservation laws or with finite differencess (FDs). A more elegant way to deal with implicit constraints is the so called adjoint approach. In this case, the constraints are added to the cost function via Lagrangian multipliers leading to the so called Lagrangian. A variation of the Lagrangian with respect to the state variables leads to the adjoint equations. Solving these adjoint equations, the derivative of the cost functional can be computed with a computational effort independent of the number of control parameters. In principle, adjoint based optimization is capable of identifying an optimal control even for highly non-linear and time-dependent control problems. The solution of the adjoint system can be computationally and implementationally complex, however. One reason for this is that for non-linear systems the complete state space must be known to obtain the adjoint solution. As the adjoint equations have to be solved backwards in time this constitutes a severe memory demand for time dependent problems. Furthermore, several approaches for the implementation of the adjoint system exist numerically, which differ considerably in terms of accuracy, implementational effort and computational cost. An important example is the choice of whether the adjoint system is derived before or after the discretization of the partial differential equation (PDE), leading to the so called continuous or discrete adjoint, respectively. The computational cost and the requirement of knowing the complete state space of the adjoint method makes it impractical for experimental applications. Nevertheless, the knowledge of the optimal control might help to identify effective control mechanisms, for example using the compressed sensing approach [64].

Noise reduction constitutes a challenging subtopic in the area of flow control due to the large separation of scales between the near field involving turbulent motion and the noise radiated into the farfield. Many investigations were conducted to connect the farfield noise with the turbulent motion found in the near field noise emitting area [25, 13, 44, 80, 95, 51, 92, 49] and to identify noise sources [31, 29, 32, 83, 36]. Nevertheless, there is still a lack of simple, yet sufficient models for typical flow control applications, which could yield simplified models for noise control.The separation of scales and the still incomplete understanding of noise generating mechanisms makes the adjoint method an attractive method for noise minimization, which is the topic of this work.

1.2. Review of Previous Work

This section will overview some of the previous work in the field of adjoint based flow control. In [22] and [5] a channel flow could be relaminarized by controlling the turbulent kinetic energy using adjoint method for direct numerical simulation (DNS) and large eddy simulation (LES). Amongst other things, it could be observed that the efficiency of the control framework decreased with increasing Reynolds number and control interval length. In a similar approach the turbulent kinetic energy of two dimensional counter rotating vortices interacting with a wall was controlled via boundary suction in [23].

In [103, 104] the continuous adjoint was used to control the noise of a two dimensional shear layer. This work constitute one of the first examples of the application of adjoint based optimization for noise minimization. In [84] the discrete adjoint approach was used to investigate the noise sensitivity of a two dimensional mixing layer. The noise emission of three dimensional round [50] and plane [59] jets was successfully reduced using the continuous adjoint. In order to circumvent the computationally expensive direct simulation of the farfield a flow/adjoint solver was coupled with a Ffowcs Williams and Hawkins approach in [78]. This way the shape of a two dimensional NACA 0012 profile could be optimized to minimize noise radiation.

The continuous adjoint was successfully applied to optimize the lift coefficient of different profiles on unsteady and unstructured grids in [1, 16]. The two dimensional discrete adjoint equations were used for shape optimization in [19] and different approximations to the discrete adjoint were tested. [75] used automatic differentiation (AD) to compute the adjoint for a time dependent two dimensional flow configuration. In [68] a shape optimization for a tiltrotor configuration was performed. This is one of the first simulations computing the discrete adjoint for a three dimensional unsteady setup. [20] compared adjoint eigenmodes obtained using the continuous and discrete adjoint of a two dimensional jet using a low Mach number code.

Adjoint optimization involves a computationally expensive iteration. Furthermore, the flow field reconstruction during the adjoint solution can demand severe hardware requirements. Because of this, instationary adjoint solutions have largely been restricted to two dimensional flows or rather small grids (e.g. $\sim 2 \cdot 10^6$ grid points in [22, 50]). The largest continuous adjoint simulation known to the authors is reported in [82], which comprised $33.6 \cdot 10^6$ grid points. In [68] the discrete adjoint was computed on a grid with $\sim 5 \cdot 10^6$ nodes. Furthermore, questions remain regarding the accuracy of the continuous adjoint approach. This work will address these questions by computing adjoints of a plane jet with different resolutions and investigating the optimization efficiency as a function of gradient accuracy and resolution.

Second derivative information can be obtained by solving the sensitivity equations of the adjoint equations. In many cases optimization schemes exploiting information about the Hessian have an improved performance compared to purely gradient based optimization schemes. Examples for optimizations exploiting second derivative information can be found in [102, 43, 42, 77, 93]. However, these works are restricted to rather small numerical systems (at most $2 \cdot 10^4$ grid points in [77]) and a moderate number of control parameters (6400 in [42] and at most a few dozen in [102, 43, 77, 93]). Thus, an open question is if such optimization schemes are efficient for three dimensional flow cases including a high number of control parameters.

This work is organized as follows. In chapter 2 the derivative based optimization algorithms used in this work are introduced. Chapter 3 explains how to use the adjoint method to obtain first and second derivative information. Next, the numerical methods used to solve the flow and adjoint equations are explained in chapter 4. Details about the optimization framework and several numerical parameters are given in chapter 5. In chapter 6 the accuracy of different adjoint formulations is investigated in detail. The adjoint method is used for the purpose of noise minimization of plane jets in chapter 7. Finally, the results are summarized in chapter 8.

2. Unconstrained Optimization

Given a cost-functional $\Im(\mathbf{g}) : G \to \Re$ the general unconstrained optimization problem reads:

$$\text{find } \min_{\mathbf{g}\in G} \Im(\mathbf{g}) \tag{2.1}$$

In this chapter the most common optimization schemes are presented based on information about the gradient $\frac{d\Im}{d\mathbf{g}}$ and/or the Hessian $\frac{d^2\Im}{d\mathbf{g}^2}$. The discussion will give only a brief motivation of the methods and concentrates on the algorithms. There will be no complete derivation of the methods, proof of convergence etc. The reader may refer to [70] for a more detailed discussion. The conjugate gradient (CG) and Broyden-Fletcher-Goldfarb-Shanno (BFGS) methods are also explained in some detail in [74]. A summary of Newton Krylov methods can be found in [53]. The discussion of the various methods is necessary, however, since we perform a detailed comparison of the performance of these algorithms for our control problem in this work. To the best knowledge of the author, this has not been done before for an unsteady flow optimization problem .

2.1. Line Search

In line search based optimization algorithms the control is updated iteratively as

$$\mathbf{g}_{i+1} = \mathbf{g}_i + \alpha_i \mathbf{p}_i, \tag{2.2}$$

where $\mathbf{p}_i$ is some search direction, α_i is a step length and $\mathbf{g}_i$ is the control at the i^{th} iteration. Algorithms for obtaining the search direction $\mathbf{p}_i$ are given in section 2.2 and 2.3. This section deals with the problem of determining proper step lengths α_i.

Given the search direction $\mathbf{p}_i$ the step length α_i is often chosen such that the Wolfe conditions

$$\Im(\mathbf{g}_i + \alpha_i \mathbf{p}_i) \leq \Im(\mathbf{g}_i) + c_1 \alpha_i \frac{d\Im(\mathbf{g}_i)}{d\mathbf{g}} \mathbf{p}_i \tag{2.3}$$

$$\frac{d\Im(\mathbf{g}_i + \alpha_i \mathbf{p}_i)}{d\mathbf{g}} \mathbf{p}_i \geq c_2 \frac{d\Im(\mathbf{g}_i)}{d\mathbf{g}} \mathbf{p}_i \tag{2.4}$$

are satisfied for some constants c_1 and c_2 with $0 < c_1 < c_2 < 1$. The sufficient decrease Condition (2.3) ensures that the step length α_i does not become too large, as it requires a higher reduction with higher step length. On the other hand, the curvature condition (2.4) prevents the step length from becoming too short as it requires that the gradient of the new iteration point increases compared to the previous point.

It can be shown that every line search optimization following iteration (2.2) is globally convergent if α_i satisfies the Wolfe conditions and $\mathbf{p}_i$ is a descent direction for all i and if the sequence $\mathbf{p}_i$ does not become perpendicular to the gradient.

Alternatively, the strong Wolfe conditions, which also satisfy the Wolfe conditions, are used:

$$\Im(\mathbf{g}_i + \alpha_i \mathbf{p}_i) \leq \Im(\mathbf{g}_i) + c_1 \alpha_i \frac{d\Im(\mathbf{g}_i)}{d\mathbf{g}} \mathbf{p}_i \tag{2.5}$$

$$\left| \frac{d\Im(\mathbf{g}_i + \alpha_i \mathbf{p}_i)}{d\mathbf{g}} \mathbf{p}_i \right| \leq c_2 \left| \frac{d\Im(\mathbf{g}_i)}{d\mathbf{g}} \mathbf{p}_i \right| . \tag{2.6}$$

In opposition to the Wolfe conditions the strong Wolfe conditions guarantee that the step length is sufficiently near to a local minimizer in the sense that the norm of the gradient in the search direction has to decrease.

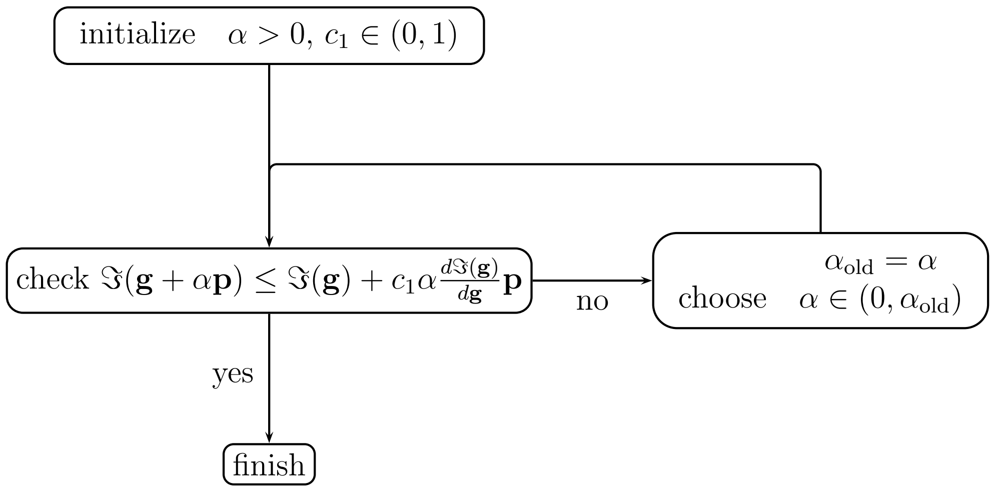
2.1.1. Backtracking

The Armijo backtracking algorithm, shown in algorithm 1, gives a strategy how to obtain a step length satisfying the Wolfe conditions. Starting with some initial step length, this step length is decreased until the sufficient decrease condition (2.5) is satisfied. Note that the curvature condition (2.4) is not explicitly tested. Instead, the backtracking algorithm 1 avoids small step lengths implicitly, by decreasing the initial step length only as much as necessary.

A prominent strategy for the updating of the step length is $\alpha = \rho\alpha_{\text{old}}$ with some $\rho \in (0, 1)$. Another method sets α to the extremum of the quadratic function defined by the function values at 0 and α_{old} and the directional derivative at 0. Note, that this extremum is a minimum in $(0, \alpha_{\text{old}})$ as long as $\mathbf{p}_i$ is a descent direction and the sufficient decrease condition (2.5) isn't fulfilled.

2.1.2. Exact Line Search

Given some step lengths $\alpha < \beta < \gamma$ it can easily be shown that a local minimizer of $\Im$ exists in the interval (α, γ) if $\Im(\alpha) > \Im(\beta) < \Im(\gamma)$. Here, the definition $\Im(\alpha) \equiv \Im(\mathbf{g}_i + \alpha\mathbf{p}_i)$ is introduced. Choosing a new step length $\delta \in (\alpha, \gamma)$ it is always possible to choose a triple of step lengths, depending on the function value $\Im(\delta)$, such that the new step length


initialize $\alpha > 0$, $c_1 \in (0,1)$
check $\Im(\mathbf{g}+\alpha\mathbf{p}) \leq \Im(\mathbf{g}) + c_1\alpha\frac{d\Im(\mathbf{g})}{d\mathbf{g}}\mathbf{p}$
no
$\alpha_{\text{old}} = \alpha$
choose $\alpha \in (0, \alpha_{\text{old}})$
yes
finish

Algorithm 1: Armijo backtracking algorithm for choosing a step length α such that the Wolfe conditions are satisfied.

triple contains a local minimizer and the new search interval is smaller than the previous one. Repeating this procedure the position of the local minimizer can be hunted down iteratively to an arbitrary precision.[1] In this work the Brent line minimization algorithm as described in [74] is implemented.

Exact line searches are usually quite expensive and only useful if a high precision of the line search is necessary, as in e.g. CG methods.

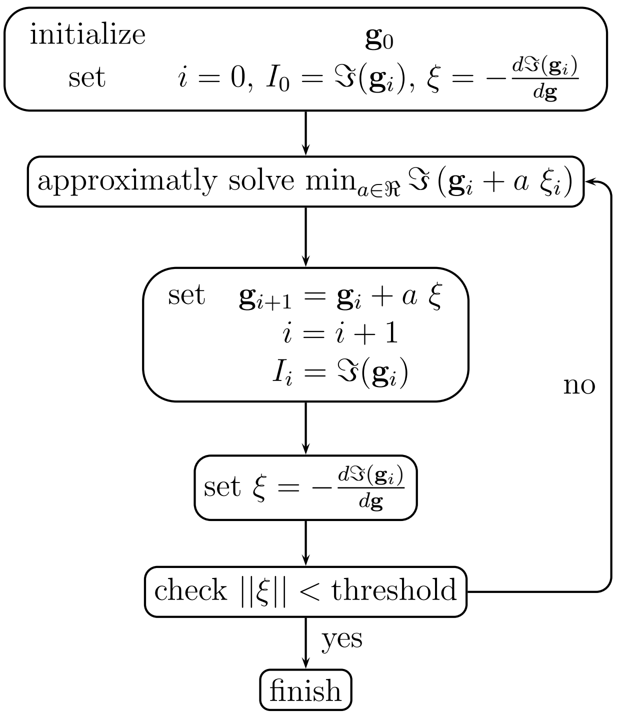
2.1.3. Wolfe Line Search

One can also try to find a step length such that the strong Wolfe conditions (2.5)-(2.6) are satisfied.

In a first phase a given step length is increased until an interval is found that, assuming a smooth function, has to contain step lengths fulfilling the strong Wolfe conditions. In the second phase the determined interval is decreased, such that the new chosen subinterval still has to contain step lengths fulfilling the strong Wolfe conditions, until a satisfying step length is found.

The implementation used in this work follows the Wolfe line search algorithm proposed in [70] with the step increase and decrease strategies given in [66].

[1]This procedure is very similar to the bisection procedure used for determining the root of a function.


initialize $\mathbf{g}_0$
set $i = 0,\ I_0 = \Im(\mathbf{g}_i),\ \xi = -\frac{d\Im(\mathbf{g}_i)}{d\mathbf{g}}$
approximately solve $\min_{a\in\Re} \Im\left(\mathbf{g}_i + a\ \xi_i\right)$
set $\mathbf{g}_{i+1} = \mathbf{g}_i + a\ \xi$
$i = i + 1$
$I_i = \Im(\mathbf{g}_i)$
set $\xi = -\frac{d\Im(\mathbf{g}_i)}{d\mathbf{g}}$
check $||\xi|| <$ threshold
no
yes
finish

Algorithm 2: The steepest descent algorithm for non-linear optimization.

2.2. Gradient Based Optimization

In this section the most common algorithms for obtaining search directions based on gradient information are introduced.

2.2.1. Steepest Descent

As the negative of the gradient of the cost-functional is the direction of steepest descent a very intuitive optimization method is to perform a line minimization along the gradient direction at every optimization iteration. This steepest descent algorithm is given in algorithm 2.

Though this algorithm seems reasonable, can be shown to be globally convergent and is very easy to implement, its performance is usually known to be poor. The reason for the often poor performance of the steepest descent algorithm is closely connected with the observation that, after an exact line minimization, the gradient at the new iteration point is perpendicular to the previous search direction. Suppose the cost function has the shape of a long narrow "valley" (ill-conditioned Hessian matrix). Then, due to the perpendicular

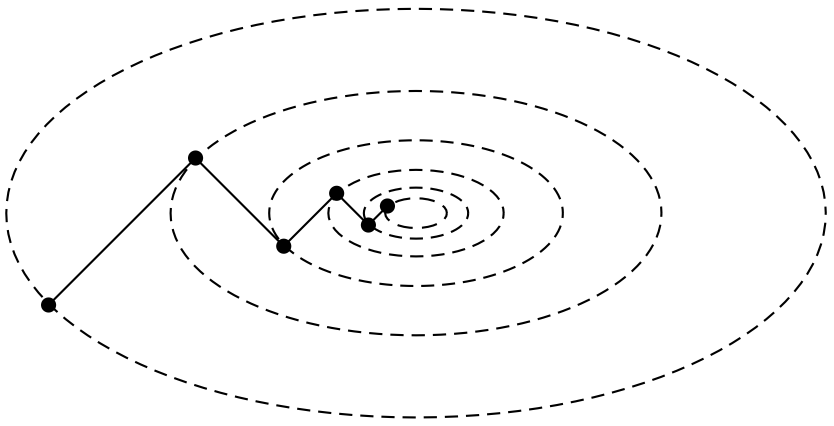


Figure 2.1: Steepest descent optimization for the cost function $\Im = x^2 + 4y^2$ with an exact line search. Dashed lines are iso-contours of the cost function and the dots show the iterations of the steepest descent algorithm. Note, that the way to the minimum reveals a zigzag pattern.

search directions, the search direction can not be adjusted along this valley. This situation is illustrated in figure 2.1. Because of this deficiency the steepest descent algorithm is not discussed further.

2.2.2. Conjugate Gradient

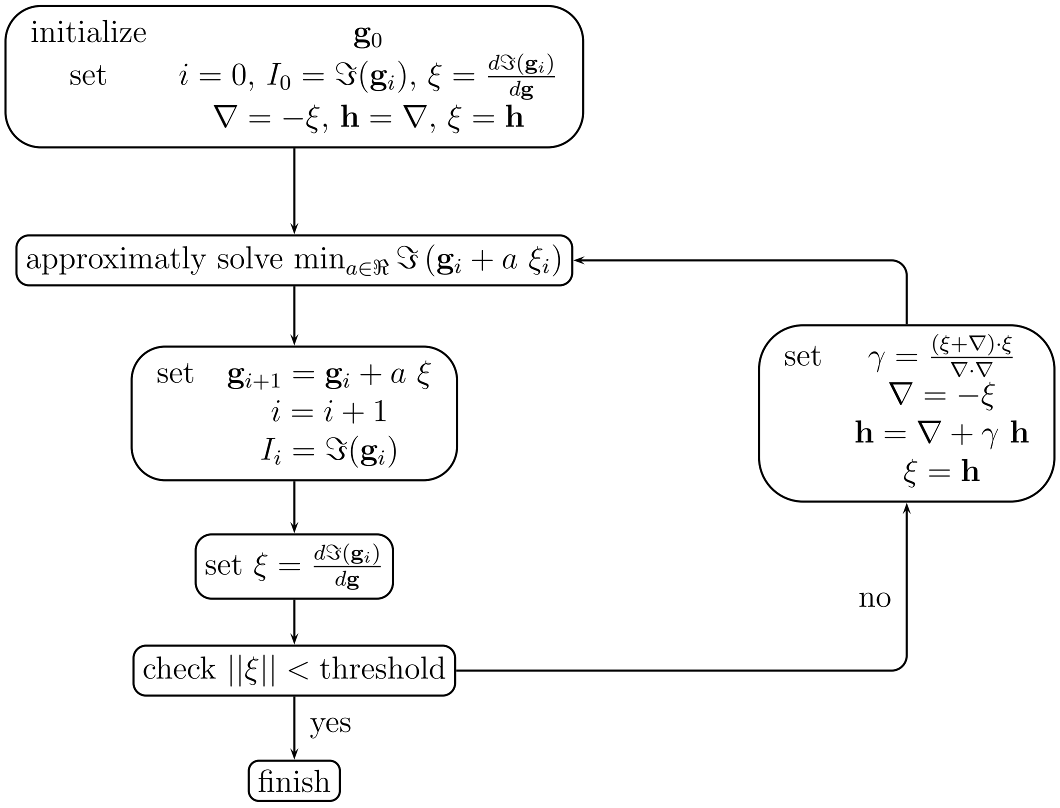
The basic idea of the CG method is to choose the new search directions in such a way that it does "not interfere" with the previous search directions. More mathematically speaking this translates to the requirement that the gradient obtained after a line minimization is perpendicular to all previous search directions. Expanding the cost functional to the second order of a Taylor series expansion this condition leads to

$$\mathbf{u}^T H \mathbf{v} \overset{!}{=} 0 \tag{2.7}$$

where $\mathbf{u}$ and $\mathbf{v}$ are search directions and H is the Hessian matrix of the cost functional. Two vectors are said to be conjugate if they fulfill equation (2.7).

What makes the idea of CG practicable is that a set of conjugate search directions can be constructed with the knowledge of the gradient of the actual and the previous iteration, only, using algorithm 3. Note, that except for the updating step of the new search direction the CG algorithm is identical to the steepest descent algorithm. Furthermore, despite the control at the actual optimization iteration step only three additional fields ξ, ∇ and $\mathbf{h}$ have to be kept in memory. Refer to [70] for a proof that algorithm 3 produces conjugate search directions.

The line minimization must be exact for the CG algorithm to produce conjugate search directions. However, it can be shown that the search directions remain to be a descent direction and global convergence is preserved if the strong Wolfe conditions are satisfied. Nevertheless, the line search should be performed rather accurate for the CG algorithm to produce improved search directions compared to the steepest descent direction.


initialize $\mathbf{g}_0$
set $i = 0,\ I_0 = \Im(\mathbf{g}_i),\ \xi = \frac{d\Im(\mathbf{g}_i)}{d\mathbf{g}}$
$\nabla = -\xi,\ \mathbf{h} = \nabla,\ \xi = \mathbf{h}$
approximatly solve $\min_{a\in\Re} \Im\left(\mathbf{g}_i + a\ \xi_i\right)$
set $\mathbf{g}_{i+1} = \mathbf{g}_i + a\ \xi$
$i = i+1$
$I_i = \Im(\mathbf{g}_i)$
set $\xi = \frac{d\Im(\mathbf{g}_i)}{d\mathbf{g}}$
check $||\xi|| <$ threshold
yes
finish
no
set $\gamma = \frac{(\xi+\nabla)\cdot\xi}{\nabla\cdot\nabla}$
$\nabla = -\xi$
$\mathbf{h} = \nabla + \gamma\ \mathbf{h}$
$\xi = \mathbf{h}$

Algorithm 3: Illustration of the conjugate gradient algorithm for non-linear optimization.

2.2.3. BFGS

The change of the gradient of the cost functional for different controls contains second derivative information. This observation leads to the idea to construct the Hessian of the cost functional with the information given by the gradients obtained during the optimization iterations. There are many schemes that explore this idea like the Broyden class or SR1. However, in this work only the BFGS scheme is considered, which is a member of the Broyden class.

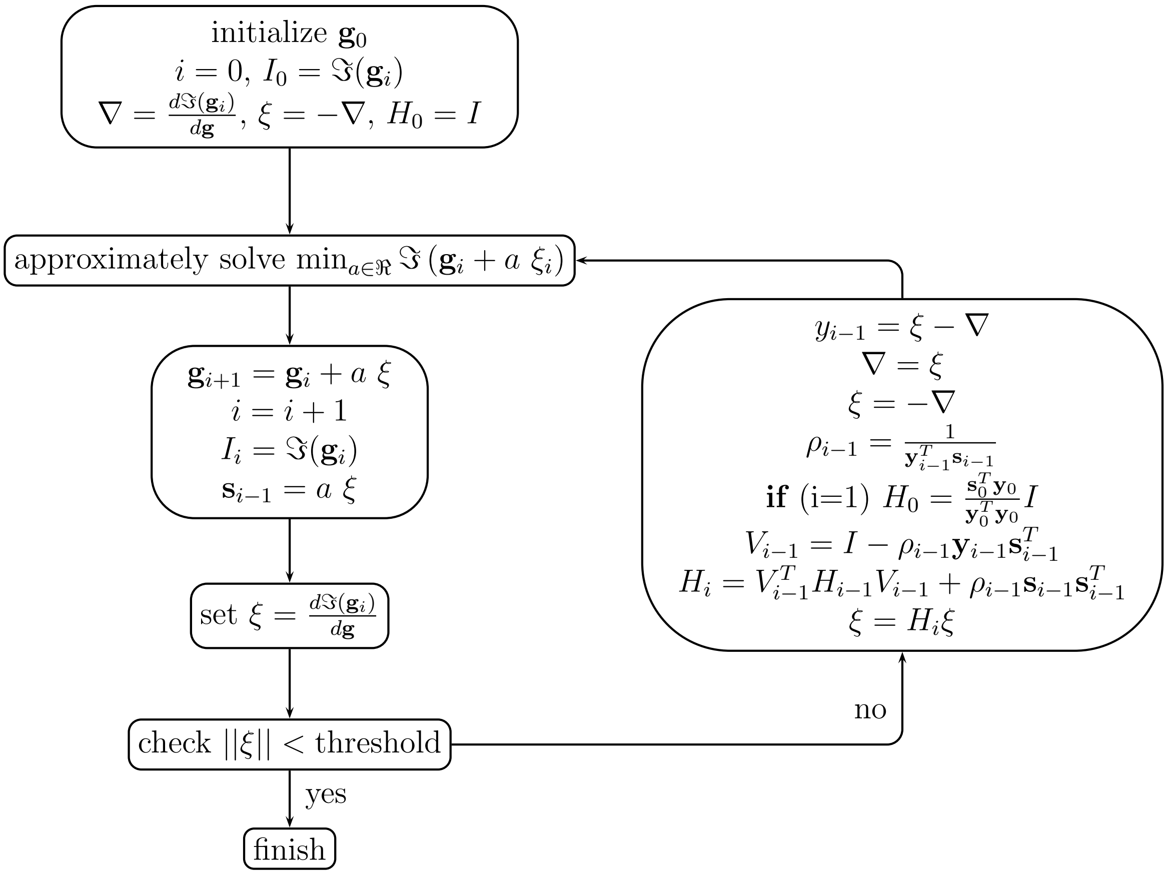
The BFGS updating formulae are given by

$$H_{k+1} = V_k^T H_k V_k + \rho_k s_k s_k^T \tag{2.8}$$

where

$$\rho_k = \frac{1}{\mathbf{y}_k^T \mathbf{s}_k} \tag{2.9}$$

$$V_k = I - \rho_k \mathbf{y}_k \mathbf{s}_k^T \tag{2.10}$$


initialize $\mathbf{g}_0$
$i = 0,\ I_0 = \Im(\mathbf{g}_i)$
$\nabla = \frac{d\Im(\mathbf{g}_i)}{d\mathbf{g}},\ \xi = -\nabla,\ H_0 = I$
approximately solve $\min_{a\in\Re} \Im\left(\mathbf{g}_i + a\ \xi_i\right)$
$\mathbf{g}_{i+1} = \mathbf{g}_i + a\ \xi$
$i = i + 1$
$I_i = \Im(\mathbf{g}_i)$
$\mathbf{s}_{i-1} = a\ \xi$
set $\xi = \frac{d\Im(\mathbf{g}_i)}{d\mathbf{g}}$
check $||\xi|| <$ threshold
yes
finish
no
$y_{i-1} = \xi - \nabla$
$\nabla = \xi$
$\xi = -\nabla$
$\rho_{i-1} = \frac{1}{\mathbf{y}_{i-1}^T \mathbf{s}_{i-1}}$
if (i=1) $H_0 = \frac{\mathbf{s}_0^T \mathbf{y}_0}{\mathbf{y}_0^T \mathbf{y}_0} I$
$V_{i-1} = I - \rho_{i-1} \mathbf{y}_{i-1} \mathbf{s}_{i-1}^T$
$H_i = V_{i-1}^T H_{i-1} V_{i-1} + \rho_{i-1} \mathbf{s}_{i-1} \mathbf{s}_{i-1}^T$
$\xi = H_i \xi$

Algorithm 4: Illustration of the BFGS algorithm for non-linear optimization.

$$\mathbf{s}_k = \mathbf{g}_{k+1} - \mathbf{g}_k \tag{2.11}$$

$$\mathbf{y}_k = \frac{d\Im(\mathbf{g}_{k+1})}{d\mathbf{g}} - \frac{d\Im(\mathbf{g}_k)}{d\mathbf{g}} \tag{2.12}$$

The matrix H_k yields an approximation to the inverse of the Hessian and thus an approximation to the Newton direction can be obtained by simply multiplying the gradient with H_k. This approximation to the Newton direction can be used for optimization.

The only question remaining is how to choose the initial matrix H_0. A common choice is to initially optimize in the gradient direction and then set $H_0 = \frac{\mathbf{s}_0^T \mathbf{y}_0}{\mathbf{y}_0^T \mathbf{y}_0} I$. The BFGS scheme together with a line minimization is shown in algorithm 4.

2.2.3.1. LBFGS

Let N be the dimension of the control space G then the matrix H_k has N^2 entries. As H_k is usually not sparse this implies that the BFGS scheme can not be directly applied for very large N as in the cases presented in this work.

```
$\mathbf{q} = \mathbf{g}$
for $m = k-1, k-2, \ldots, k-M$
    $\alpha_m = \rho_m \mathbf{s}_m^T \mathbf{q}$
    $\mathbf{q} = \mathbf{q} - \alpha_m \mathbf{y}_m$
end
$\mathbf{q} = H_k^0 q$
for $m = k-M, k-M+1, \ldots, k-1$
    $\beta = \rho_m \mathbf{y}_m^T \mathbf{q}$
    $\mathbf{q} = \mathbf{q} + (\alpha_m - \beta)\mathbf{s}_m$
end
```

Algorithm 5: The LBFGS algorithm for computing the product of a vector with the BFGS matrix without explicit usage of the matrix.

This gives rise to the necessity of low storage variants of the BFGS scheme. In this low storage BFGS (LBFGS) scheme, instead of calculating H_k directly, the vectors $\mathbf{s}_k$ and $\mathbf{y}_k$ are saved and the product of H_k with some vector is calculated by multiplying this vector with $\mathbf{s}_k$ and $\mathbf{y}_k$. To restrict the memory requirements only the last M appearances of $\mathbf{s}_k$ and $\mathbf{y}_k$ are saved and information from older iterations are discarded.

The product of $\mathbf{q} = H_k\mathbf{g}$ using the last M iterations can be computed efficiently with algorithm 5. H_k^0 is an approximation to the BFGS matrix at iteration $k - M - 1$. A common choice is $H_k^0 = \frac{\mathbf{s}_{k-1}^T \mathbf{y}_{k-1}}{\mathbf{y}_{k-1}^T \mathbf{y}_{k-1}} I$. Note, that the optimization algorithm 4 can be adopted for LBFGS by exchanging the line $\xi = H_i \xi$ with algorithm 5.

2.3. Hessian Based Optimization

In this section optimization schemes based on information of the Hessian are presented. It should be noted that the BFGS algorithm is often denoted as Hessian based in the literature as it constructs an approximation to the Hessian. It is, however, Hessian free in the sense that it does not use the exact Hessian explicitly. Note, that for a very high dimension of the control space an exact computation or inversion of the Hessian is impractical. Thus, this section is restricted to algorithms which require the computation of a Hessian vector product, only.

2.3.1. Newton Conjugate Gradient

It is well known that the Newton direction ξ_N, defined as the optimizer of the quadratic approximation of the cost function and given by

$$H\big|_{\mathbf{g}_i} \xi_N = - \left.\frac{d\Im}{d\mathbf{g}}\right|_{\mathbf{g}_i}, \tag{2.13}$$

where H is the Hessian of the cost function, is a good search direction and can lead to fast convergence. Unfortunately, the Newton direction cannot be used directly for minimization as pure Newton direction based methods are usually numerically unstable. Furthermore, the Newton direction leads to a maximum instead of a minimum for the case of negative curvature (negative second derivative) of the cost function. Additionally, the Hessian may not be invertible.

Nevertheless, the idea of the Newton direction can be exploited with the conjugate gradient for linear systems (CGLin) algorithm. This algorithm is in principle the same as the CG algorithm for non linear systems introduced in section 2.2.2. However, here the CGLin algorithm is used for the solution of the linear system 2.13 and major modifications are made to increase the efficiency of the CG algorithm for linear systems.

Minor but important modifications of the CGLin algorithm are necessary to adopt it to Newton based minimization. As stated earlier, directions with a negative curvature lead the solution away from a local minimizer. Furthermore, the CGLin algorithm converges for positive definite matrices only. Thus, the CGLin algorithm is aborted as soon as a search direction with negative curvature appears. In this case the last solution obtained by the CGLin algorithm before the occurrence of a negative curvature direction is used as a search direction for the minimization. Another modification is that the CGLin algorithm is always initialized with $\xi = 0$ to avoid negative curvatures in the solution of the CGLin algorithm. Furthermore, with this choice the first iteration gives the steepest descent direction, which is improved in later iterations of the CGLin algorithm. We will refer to the modified CGLin iteration in conjunction with a line search as the Newton conjugate gradient with line search (NCGline) iteration.

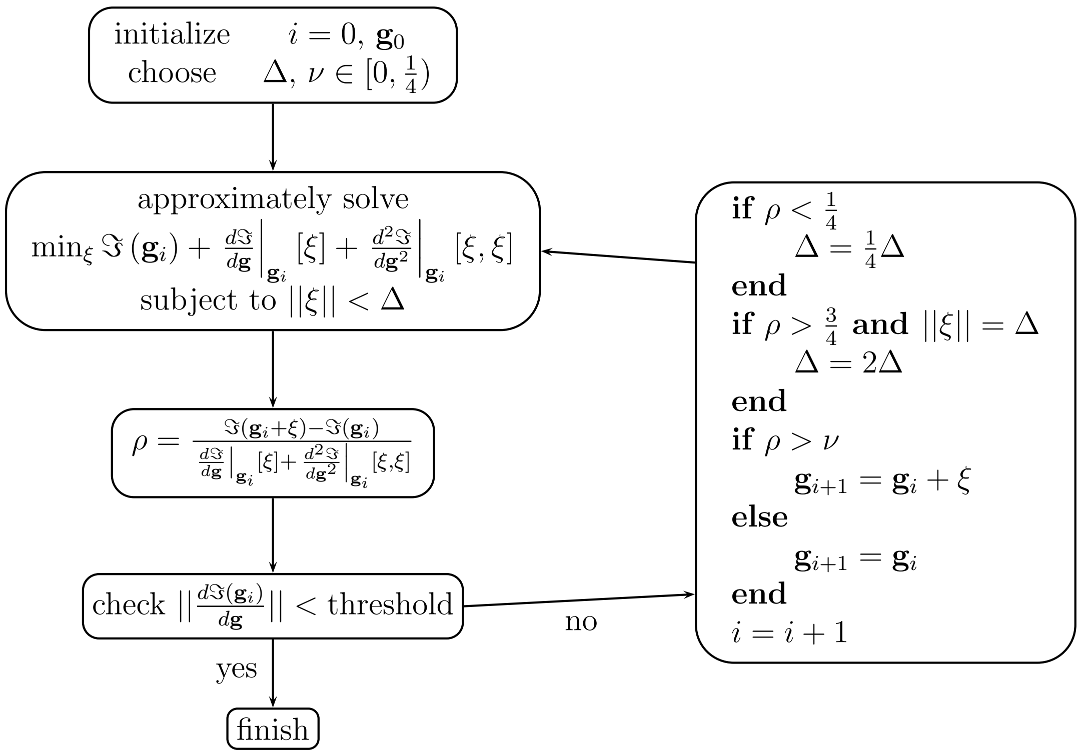
2.3.2. Trust Region Optimizations

An alternative to the line search based minimization techniques are the so called trust region algorithms [93]. The variant of the trust region algorithm implemented for this work is depicted in algorithm 6. The basic idea is that, given some trust region Δ, one tries to find an approximate solution to the trust region subproblem

$$\min_{\xi} \left(\left. \frac{d\Im}{d\mathbf{g}} \right|_{\mathbf{g}_i} (\xi) + \left. \frac{d^2\Im}{d\mathbf{g}^2} \right|_{\mathbf{g}_i} (\xi, \xi) \right) \quad \text{subject to} \quad ||\xi|| < \Delta, \tag{2.14}$$

that is the minimizer ξ to the quadratic model of the cost function in the region that satisfies $||\xi|| \leq \Delta$. This quadratic minimizer is used as a new trial point for the minimization of the cost function.

The trust region Δ must not be chosen too small to ensure an efficient behavior of the minimization iteration. On the other hand it must be small enough such that the quadratic approximation to the cost function is valid. Because of this, at every iteration the new function value is compared to the value obtained by the quadratic model and the trust region is updated depending on the agreement between real cost function value and quadratic


initialize $i = 0$, $\mathbf{g}_0$
choose Δ, $\nu \in [0, \frac{1}{4})$
approximately solve
$\min_\xi \Im(\mathbf{g}_i) + \frac{d\Im}{d\mathbf{g}}\Big|_{\mathbf{g}_i}[\xi] + \frac{d^2\Im}{d\mathbf{g}^2}\Big|_{\mathbf{g}_i}[\xi,\xi]$
subject to $||\xi|| < \Delta$
$\rho = \frac{\Im(\mathbf{g}_i+\xi)-\Im(\mathbf{g}_i)}{\frac{d\Im}{d\mathbf{g}}\big|_{\mathbf{g}_i}[\xi]+\frac{d^2\Im}{d\mathbf{g}^2}\big|_{\mathbf{g}_i}[\xi,\xi]}$
check $||\frac{d\Im(\mathbf{g}_i)}{d\mathbf{g}}|| <$ threshold
yes
finish
no
if $\rho < \frac{1}{4}$
$\Delta = \frac{1}{4}\Delta$
end
if $\rho > \frac{3}{4}$ **and** $||\xi|| = \Delta$
$\Delta = 2\Delta$
end
if $\rho > \nu$
$\mathbf{g}_{i+1} = \mathbf{g}_i + \xi$
else
$\mathbf{g}_{i+1} = \mathbf{g}_i$
end
$i = i + 1$

Algorithm 6: The trustregion algorithm.

model. If the actual reduction of the cost function is close to or even higher than the quadratic approximation guess, the trust region is increased for the next iteration. If the cost function reduction is small (or negative) the new search point is rejected and the trust region is decreased.

In most cases obtaining the exact solution to the trust region subproblem (2.14) is difficult and one must stick with an approximation. Furthermore, for very high dimensional control spaces even obtaining the exact solution of the Newton equation (2.13) may be to expensive. Therefore, approximate solution schemes to equation (2.14) like the dog leg or two dimensional subspace methods, which require at least one solution to equation (2.13), can not be used for our purposes and are not explained any further. The CGLin algorithm can be modified to yield an approximation to the trust region subproblem (2.14). The first observation is that the choice $\xi = 0$ in the CGLin algorithm ensures that the solution vector is monotonically increasing with iteration number. Thus, it can be stated that if the solution vector is outside the trust region for some iteration i it will be outside the trust region for all iterations $j > i$. Hence, one might stop the CGLin iteration immediately, as soon as the solution vector reaches the trust region and follow the last search direction only to the trust region boundary. In case of directions of negative curvature the iteration is stopped and one follows the last search direction to the trust region boundary. The modified CGLin iteration in conjunction with a trust region will be referred to as the

Newton conjugate gradient with trust region (NCGtrust) iteration. Note, that in contrast to the NCGline algorithm in case of a negative curvature direction the information of this direction is not discarded.

An alternative to the NCGtrust iteration is based on the observation that a basis to the Krylov subspace of the Hessian Matrix is constructed during the CGLin iteration. So instead of aborting the iteration when negative curvature is encountered or the solution vector leaves the trust region, the iteration is resumed and the subproblem (2.14) is solved exactly in the Krylov subspace obtained by the iterations so far. This is realized with the Lanczos algorithm. This algorithm is very similar to the CGLin algorithm. However, in contrast to the CGLin algorithm the Lanczos algorithm does not break down in case of a search direction with negative curvature. We will refer to the trust region optimization in conjunction with Lanczos iteration as the Newton Lanczos (NLan) algorithm and details are given in [37].

3. Constraint Optimization with Adjoint Method

Given some state variables $\mathbf{\Phi} \in P$, some control $\mathbf{g} \in G$, a cost functional $\Im(\mathbf{\Phi}, \mathbf{g}) : P \times G \rightarrow \Re$ and some constraints $\mathbf{F}(\mathbf{\Phi}, \mathbf{g}) = \vec{0}$ with $\mathbf{F}(\mathbf{\Phi}, \mathbf{g}) : P \times G \rightarrow V$ the general constraint optimization problem reads:

$$\text{find } \min_{\substack{\mathbf{\Phi} \in P \\ \mathbf{g} \in G}} \Im(\mathbf{\Phi}, \mathbf{g}) \quad \text{such that } \mathbf{F}(\mathbf{\Phi}, \mathbf{g}) = 0 \tag{3.1}$$

In the following we will discuss the case where there are no constraints on the control $\mathbf{g}$, such that the state variables and the cost functional become functions of control $\mathbf{g}$, only:

$$\begin{aligned} \mathbf{\Phi} &\equiv \mathbf{\Phi}(\mathbf{g}) & (3.2) \\ \Im &\equiv \Im(\mathbf{g}) \equiv \Im(\Phi(\mathbf{g}), \mathbf{g}) & (3.3) \end{aligned}$$

However, one should keep in mind that $\mathbf{\Phi}(\mathbf{g})$ and $\Im(\mathbf{g})$ are just formal representations, as the functional dependencies are given implicitly through the constraints $\mathbf{F}(\mathbf{\Phi}, \mathbf{g}) = 0$.

3.1. Gradient through Adjoint Method

In order to understand the control approach applied later in this work, an efficient way of calculating the gradient of the cost functional with implicit constraints is presented in this section. The following is a short introduction, only. Refer to [40] for a more detailed introduction and further insight or [46] for a more rigorous mathematical treatment.

Given some control $\mathbf{g}_0$ the variation of the cost functional with respect to some variation $\mathbf{g}'$ of the control is given by:

$$\begin{aligned}\left.\frac{d\Im}{d\mathbf{g}}\right|_{\mathbf{g}_0}[\mathbf{g}'] &\equiv \lim_{\epsilon\to 0}\frac{\Im\left(\mathbf{\Phi}(\mathbf{g}_0+\epsilon\mathbf{g}'),\mathbf{g}_0+\epsilon\mathbf{g}'\right)-\Im\left(\mathbf{\Phi}(\mathbf{g}_0),\mathbf{g}_0\right)}{\epsilon}\\ &= \frac{\partial\Im}{\partial\mathbf{\Phi}}\left[\left.\frac{d\mathbf{\Phi}}{d\mathbf{g}}\right|_{\mathbf{g}_0}[\mathbf{g}']\right]+\frac{\partial\Im}{\partial\mathbf{g}}[\mathbf{g}']\,.\end{aligned} \tag{3.4}$$

To simplify the notation $\frac{d\Im}{d\mathbf{g}}$ will be written in the following instead of $\frac{d\Im}{d\mathbf{g}}|_{\mathbf{g}=\mathbf{g}_0}$. The variation of the state variables or sensitivity $\mathbf{\Phi}' = \frac{d\mathbf{\Phi}}{d\mathbf{g}}[\mathbf{g}']$ can be determined with the so called sensitivity equation, given by the variation of the constraints $\mathbf{F}$:

$$\frac{d\mathbf{F}}{d\mathbf{g}}[\mathbf{g}'] = \frac{\partial\mathbf{F}}{\partial\mathbf{\Phi}}\left[\frac{d\mathbf{\Phi}}{d\mathbf{g}}[\mathbf{g}']\right]+\frac{\partial\mathbf{F}}{\partial\mathbf{g}}[\mathbf{g}'] = \frac{\partial\mathbf{F}}{\partial\mathbf{\Phi}}[\mathbf{\Phi}']+\frac{\partial\mathbf{F}}{\partial\mathbf{g}}[\mathbf{g}'] = 0. \tag{3.5}$$

However, determining the gradient $\frac{d\Im}{d\mathbf{g}}$ this way is not feasible for high dimensional control spaces, as equations (3.4) and (3.5) have to be solved N times, if N is the dimension of the control space.

This expensive calculation of $\frac{d\Im}{d\mathbf{g}}$ can be circumvented by introducing the so called Lagrangian multipliers $\xi \in V$, which are defined as the solution of

$$\frac{\partial\Im}{\partial\mathbf{\Phi}}[\mathbf{\Phi}'] - \left\langle \xi, \frac{\partial\mathbf{F}}{\partial\mathbf{\Phi}}[\mathbf{\Phi}']\right\rangle_V = 0 \qquad \forall\ \mathbf{\Phi}' \in P, \tag{3.6}$$

where $\langle\cdot,\cdot\rangle_X$ is a scalar product in the vector space X.

Substituting equations (3.6) and (3.5) into equation (3.4) one gets:

$$\begin{aligned}\frac{d\Im}{d\mathbf{g}}[\mathbf{g}'] &= \frac{\partial\Im}{\partial\mathbf{\Phi}}[\mathbf{\Phi}']+\frac{\partial\Im}{\partial\mathbf{g}}[\mathbf{g}']\\ &\overset{(3.6)}{=} \left\langle \xi, \frac{\partial\mathbf{F}}{\partial\mathbf{\Phi}}[\mathbf{\Phi}']\right\rangle_V + \frac{\partial\Im}{\partial\mathbf{g}}[\mathbf{g}']\\ &\overset{(3.5)}{=} -\left\langle \xi, \frac{\partial\mathbf{F}}{\partial\mathbf{g}}[\mathbf{g}']\right\rangle_V + \frac{\partial\Im}{\partial\mathbf{g}}[\mathbf{g}']\,.\end{aligned} \tag{3.7}$$

Note, that in equation (3.7) the variation of the state variables $\mathbf{\Phi}'$ has vanished and thus the gradient $\frac{d\Im}{d\mathbf{g}}$ can be obtained directly if the adjoint variables ξ are known. At first sight this does not seem to be an advantage as equation (3.6) does still contain $\mathbf{\Phi}'$.

Thus, we introduce the adjoint A^* of an operator A with the definition

$$\langle \mathbf{x}, A[\mathbf{y}]\rangle_X \overset{!}{=} \langle \mathbf{y}, A^*[\mathbf{x}]\rangle_Y \quad \forall\ \mathbf{x}\in X, \mathbf{y}\in Y. \tag{3.8}$$

Using this definition equation (3.6) can be rewritten as

$$\begin{aligned}
&\left\langle 1, \frac{\partial \Im}{\partial \mathbf{\Phi}} [\mathbf{\Phi}'] \right\rangle_{\Re} - \left\langle \xi, \frac{\partial \mathbf{F}}{\partial \mathbf{\Phi}} [\mathbf{\Phi}'] \right\rangle_{V} \\
&= \left\langle \mathbf{\Phi}', \left(\frac{\partial \Im}{\partial \mathbf{\Phi}}\right)^{*} [1] \right\rangle_{P} - \left\langle \mathbf{\Phi}', \left(\frac{\partial \mathbf{F}}{\partial \mathbf{\Phi}}\right)^{*} [\xi] \right\rangle_{P} \\
&= \left\langle \mathbf{\Phi}', \left(\frac{\partial \Im}{\partial \mathbf{\Phi}}\right)^{*} [1] - \left(\frac{\partial \mathbf{F}}{\partial \mathbf{\Phi}}\right)^{*} [\xi] \right\rangle_{P} = 0.
\end{aligned} \tag{3.9}$$

As the variation $\mathbf{\Phi}'$ is arbitrary not only the whole expression but every part in the scalar product has to be zero, leading to the so called adjoint equations

$$\left(\frac{\partial \Im}{\partial \mathbf{\Phi}}\right)^{*} - \left(\frac{\partial \mathbf{F}}{\partial \mathbf{\Phi}}\right)^{*} [\xi] = 0. \tag{3.10}$$

Using equation (3.10) to determine the adjoint variables ξ and equation (3.7) to determine the gradient $\frac{d\Im}{d\mathbf{g}}$, the computational cost for determining the gradient is now independent of the dimension of the control space G. The adjoint system has therefore to be solved only once and can be used for every change in the control variables.

3.1.1. Lagrangian

The relations illuminated in section 3.1 can be presented in a more compact way by defining the Lagrangian as

$$L \equiv \Im - \langle \xi, F \rangle_{V}. \tag{3.11}$$

Setting the partial derivatives of the Lagrangian with respect to $\mathbf{\Phi}$, $\mathbf{g}$ and ξ to zero one gets

$$\frac{\partial L}{\partial \xi} = -\mathbf{F} = 0 \tag{3.12}$$

$$\frac{\partial L}{\partial \mathbf{\Phi}} = \frac{\partial \Im}{\partial \mathbf{\Phi}} - \left\langle \xi, \frac{\partial \mathbf{F}}{\partial \mathbf{\Phi}} \right\rangle_{V} = 0 \tag{3.13}$$

$$\frac{\partial L}{\partial \mathbf{g}} = \frac{\partial \Im}{\partial \mathbf{g}} - \left\langle \xi, \frac{\partial \mathbf{F}}{\partial \mathbf{g}} \right\rangle_{V} \overset{3.7}{=} \frac{d\Im}{d\mathbf{g}} = 0. \tag{3.14}$$

It can be observed that equation (3.12) are the constraints, equation (3.13) are the adjoint equations (see equation (3.6)) and equation (3.14) is the gradient of the cost functional (see equation (3.7)). With these considerations it is evident that the constraint optimization problem (3.1) is equivalent to

$$\frac{\partial L}{\partial (\mathbf{\Phi}, \mathbf{g}, \xi)} = 0. \tag{3.15}$$

In equation (3.15) only partial derivatives are present, so it can be recognized that with the introduction of the Lagrangian multipliers the constraint optimization problem (3.1) is transformed into the system of equations (3.15).

3.1.2. Example

In this section the explicit mathematical procedure for obtaining the adjoint equations in continuous space is presented. The chosen example is the viscid one dimensional burgers equation with some forcing/control u

$$\frac{\partial y(x,t)}{\partial t} + y(x,t)\frac{\partial y(x,t)}{\partial x} = \nu \frac{\partial^2 y(x,t)}{\partial x^2} + u(x,t) \quad (x,t) \in (x_0,x_1)\times(t_0,t_1) \tag{3.16}$$

with boundary conditions

$$y(x_0,t) = y(x_1,t) = 0 \quad t \in (t_0,t_1) \tag{3.17}$$
$$y(x,0) = y_0(x) \quad x \in (x_0,x_1). \tag{3.18}$$

The goal is to choose the control $u(x,t)$ such that the state variable $y(x,t)$ matches a given function $\Upsilon(x,t)$ best as possible, thus the cost functional is defined as

$$\Im = \int_{x_0}^{x_1} dx \int_{t_0}^{t_1} dt \, (y(x,t) - \Upsilon(x,t))^2 . \tag{3.19}$$

To simplify the notation the dependencies of the functions will be skipped from now on and we will write y instead of $y(x,t)$. With the above definitions and the L_2-norm the Lagrangian is

$$\begin{aligned} L = & \int_{x_0}^{x_1} dx \int_{t_0}^{t_1} dt \left[(y-\Upsilon)^2 \right. \\ & \left. -\lambda \frac{\partial y}{\partial t} - \lambda y \frac{\partial y}{\partial x} + \lambda \nu \frac{\partial^2 y}{\partial x^2} + \lambda u \right] \\ & - \int_{t_0}^{t_1} dt \, \lambda_{bc} y \big|_{x_0,x_1} - \int_{x_0}^{x_1} dx \, \lambda_{init} (y - y_0)\big|_{t_0} , \end{aligned} \tag{3.20}$$

where λ, λ_{bc} and λ_{init} are Lagrangian multipliers. The variation of the Lagrangian with respect to y is

$$\begin{aligned} \frac{\partial L(y,u,\lambda)}{\partial y}[\widetilde{y}] \equiv & \lim_{\epsilon\to 0} \frac{L(y+\epsilon\widetilde{y},u,\lambda) - L(y,u,\lambda)}{\epsilon} \\ = & \int_{x_0}^{x_1} dx \int_{t_0}^{t_1} dt \left[2(y-\Upsilon)\widetilde{y} \right. \\ & \left. -\lambda \frac{\partial \widetilde{y}}{\partial t} - \lambda \widetilde{y} \frac{\partial y}{\partial x} - \lambda y \frac{\partial \widetilde{y}}{\partial x} + \lambda \nu \frac{\partial^2 \widetilde{y}}{\partial x^2} \right] \\ & - \int_{t_0}^{t_1} dt \, \lambda_{bc} \widetilde{y} \big|_{x_0,x_1} - \int_{x_0}^{x_1} dx \, \lambda_{init} \widetilde{y} \big|_{t_0} = 0. \end{aligned} \tag{3.21}$$

In order to obtain the adjoint equations we have to calculate the adjoint of equation (3.21). This is achieved by integration by parts and leads to

$$\begin{aligned}\frac{\partial L(y,u,\lambda)}{\partial y}[\widetilde{y}] = &\int_{x_0}^{x_1} dx \int_{t_0}^{t_1} dt \widetilde{y}\,[2\,(y-\Upsilon) \\ &+\frac{\partial \lambda}{\partial t} - \lambda\frac{\partial y}{\partial x} + \frac{\partial \lambda y}{\partial x} + \nu\frac{\partial^2 \lambda}{\partial x^2}\Bigg] \\ &-\int_{t_0}^{t_1} dt \left[\widetilde{y}\left(\lambda_{bc} + \lambda y + \nu\frac{\partial \lambda}{\partial x}\right) - \frac{\partial \widetilde{y}}{\partial x}\nu\lambda\right]_{x_0,x_1} \\ &+\int_{x_0}^{x_1} dx\,[\widetilde{y}\,(\lambda - \lambda_{init})]_{t_0} \\ &-\int_{x_0}^{x_1} dx\,\widetilde{y}\lambda|_{t_1} = 0.\end{aligned} \tag{3.22}$$

As the direction/variation $\widetilde{y}$ can be chosen arbitrarily[1] we can conclude that the different parts under the integrals itself must be zero. So the adjoint equations become

$$2\,(y-\Upsilon) + \frac{\partial \lambda}{\partial t} - \lambda\frac{\partial y}{\partial x} + \frac{\partial \lambda y}{\partial x} + \nu\frac{\partial^2 \lambda}{\partial x^2} = 0 \quad (x,t) \in (x_0,x_1) \times (t_0,t_1) \tag{3.23}$$

$$\lambda = 0 \quad x \in (x_0,x_1), t = t_1 \tag{3.24}$$

$$\nu\lambda = 0 \quad x \in \{x_0,x_1\}, t \in (t_0,t_1) \tag{3.25}$$

$$\lambda - \lambda_{init} = 0 \quad x \in (x_0,x_1), t = t_0 \tag{3.26}$$

$$\lambda_{bc} + \lambda y + \nu\frac{\partial \lambda}{\partial x} = 0 \quad x \in \{x_0,x_1\}, t \in (t_0,t_1). \tag{3.27}$$

Note, that the Lagrangian multipliers λ_{bc} and λ_{init} appear only in equation (3.26) and (3.27). So these equations can always be fulfilled and can be skipped if one is interested in λ, only. λ is determined by equations (3.23)-(3.25).

The variation of the Lagrangian with respect to the control gives

$$\begin{aligned}\frac{\partial L(y,u,\lambda)}{\partial u}[\widetilde{u}] \equiv \lim_{\epsilon\to 0}\frac{L(y,u+\epsilon\widetilde{u},\lambda) - L(y,u,\lambda)}{\epsilon} &= \int_{x_0}^{x_1} dx \int_{t_0}^{t_1} dt\widetilde{u}\lambda \\ &= \left.\frac{d\Im}{du}\right|_u [\widetilde{u}],\end{aligned} \tag{3.28}$$

which leads to the conclusion

$$\left.\frac{d\Im}{du}\right|_u = \lambda \tag{3.29}$$

[1]And at the boundaries the variation and its derivative can be chosen independently.

I/O-Library	MPI collective	MPI non-collective	fortran intrinsic binary I/O	HDF5 collective
time/s	3.3	150	3.0	12.0

Table 3.1: Performance of I/O of different libraries tested on JUGENE for the writing of one flow field comprising 148 Mio. grid points (2.96 GB).

#comp.-proc.	#I/O-proc.	I/O-freq	t_{iter}
4096	0	200	2.5
4096	0	3	6.6
4000	96	200	2.6
4000	96	3	2.8

Table 3.2: Number of compute processes, number of I/O-processes, number of flow integrations between savings of flowfield and time of one RK integration for different configurations. These performance tests were performed on JUGENE with a flow case comprising 148 Mio. grid points.

3.1.3. I/O Issue

From equation (3.24) one can see that there is a boundary condition given at final time $t = t_1$, which implies that the adjoint equations (3.23)-(3.25) must be solved backward in time. Because of the non-linearity of equation (3.16) the state variable y still appears in the adjoint equation (3.23). As the equations that determine y are solved forward in time this implies that y has to be saved or recomputed to be available for the backward in time determination of the adjoint variable λ. A repeated recomputation of the complete flow interval during the adjoint solution is too expensive. On the other hand for most practical applications the storage requirement for saving the complete flow information is too high to be kept in memory. This holds true even if hard-disk memory is used, too. Thus, a mixed approach is adopted for this work, where the flow fields are written to/read from hard disk after a fix number of Runge-Kutta (RK) iterations. The flow fields in-between are either interpolated with a third order accurate interpolation scheme or recomputed on these shorter subintervals. Note, that these subintervals should be short enough, such that the complete flow field information of this interval can be kept in the working memory (RAM). Special checkpointing strategies exist [38] to find an optimal compromise between storage requirements and saving/loading/recomputation of variables. However, for the cases considered in this work enough memory was available to implement the simple checkpointing technique of recomputation for subintervals with constant length. Note, that this strategy is the most efficient in terms computational time as no flow information has to be recomputed more than once.

Due to the usually low bandwidth of hard disk I/O the frequent saving/loading can significantly reduce the performance of the computations. Thus, special I/O-nodes were reserved to perform the I/O-tasks asynchronously to the flow computations. The idea is that the compute nodes, responsible for the flow solution, send/receive the flow fields non-blocking to/from the I/O-nodes, while the IO-nodes actually read/write the data from/to hard disk. Furthermore, each process writes its data into one separate binary file. This way no communication between the processes is necessary for the I/O and the file handles have to be opened and closed only once at the beginning and end of the simulation. This kind of I/O proved to be more efficient than hard disk access with parallel-I/O libraries such as HDF 5 or MPI as can be seen in table 3.1, which lists the required time for writing a flow field comprising 148 Mio. grid points leading to a data size of 2.96 GB (single precision). The tests were performed on cluster JUGENE of Forschungszentrum Jülich. Similar findings are reported in [68]. The effectiveness of the asynchronous I/O approach can be observed in table 3.2, where the actual computation time is listed for different numbers of compute and I/O nodes and different saving frequencies. It becomes apparent that with the asynchronous I/O only a minimal loss of efficiency can be observed.

3.2. Hessian through Adjoint Method

With the adjoint method, introduced in section 3.1, gradient information can be obtained. The first derivative gives information about the linear response of the cost functional, only. As already discussed in section 2.3 second derivative/quadratic information can also be used to yield efficient search directions. The question is if the additional implementational and computational effort for determining second derivatives is compensated by the (eventually) improved search direction. However, the efficiency of different optimization schemes vary considerably for different test cases. To be able to study the efficiency for Hessian based optimization schemes for the systems at hand a way of obtaining second order information is introduced now.

The second derivative with respect to two control variations $\mathbf{g}'$ and $\mathbf{g}^\circ$ is

$$\begin{aligned}
&\left\langle 1, \frac{d^2\Im}{d\mathbf{g}^2}[\mathbf{g}^\circ, \mathbf{g}']\right\rangle_{\Re} \equiv \left\langle 1, \frac{d\left(\frac{d\Im}{d\mathbf{g}}[\mathbf{g}']\right)}{d\mathbf{g}}[\mathbf{g}^\circ]\right\rangle_{\Re} \\
&= \left\langle 1, \frac{d\left(\frac{d\Im}{d\mathbf{g}}[\mathbf{g}^\circ]\right)}{d\mathbf{g}}[\mathbf{g}']\right\rangle_{\Re} \overset{(3.14)}{=} \left\langle 1, \frac{d\left(\frac{\partial L}{\partial \mathbf{g}}[\mathbf{g}^\circ]\right)}{d\mathbf{g}}[\mathbf{g}']\right\rangle_{\Re} \\
&= \left\langle 1, \frac{\partial\left(\frac{\partial L}{\partial \mathbf{g}}[\mathbf{g}^\circ]\right)}{\partial \mathbf{\Phi}}\left[\frac{d\mathbf{\Phi}}{d\mathbf{g}}[\mathbf{g}']\right] + \frac{\partial\left(\frac{\partial L}{\partial \mathbf{g}}[\mathbf{g}^\circ]\right)}{\partial \mathbf{g}}[\mathbf{g}'] + \frac{\partial\left(\frac{\partial L}{\partial \mathbf{g}}[\mathbf{g}^\circ]\right)}{\partial \xi}\left[\frac{d\xi}{d\mathbf{g}}[\mathbf{g}']\right]\right\rangle_{\Re} \\
&= \left\langle 1, \frac{\partial\left(\frac{\partial L}{\partial \mathbf{g}}[\mathbf{g}^\circ]\right)}{\partial \mathbf{\Phi}}[\mathbf{\Phi}'] + \frac{\partial\left(\frac{\partial L}{\partial \mathbf{g}}[\mathbf{g}^\circ]\right)}{\partial \mathbf{g}}[\mathbf{g}'] + \frac{\partial\left(\frac{\partial L}{\partial \mathbf{g}}[\mathbf{g}^\circ]\right)}{\partial \xi}[\xi']\right\rangle_{\Re}
\end{aligned} \tag{3.30}$$

$$= \left\langle \mathbf{g}^\circ, \frac{\partial\left(\left(\frac{\partial L}{\partial \mathbf{g}}\right)^*[1]\right)}{\partial \mathbf{\Phi}}[\mathbf{\Phi}'] + \frac{\partial\left(\left(\frac{\partial L}{\partial \mathbf{g}}\right)^*[1]\right)}{\partial \mathbf{g}}[\mathbf{g}'] + \frac{\partial\left(\left(\frac{\partial L}{\partial \mathbf{g}}\right)^*[1]\right)}{\partial \xi}[\xi'] \right\rangle_G .$$

Note, that analogous to the procedure in chapter 3.1, the adjoint of the operators were introduced to get rid of the dependency of the operators for some variation. From equation (3.30) one can conclude that the second part of the scalar product equals an Hessian vector product.

As the variation ξ' of the Lagrangian multiplier ξ appears in equation (3.1) an equation is needed for its determination. This equation is obtained by a variation of the adjoint equation (3.10) with respect to the control $\mathbf{g}$ [2]

$$\begin{aligned}
&\left\langle 1, \frac{d\left(\frac{\partial L}{\partial \mathbf{\Phi}}[\mathbf{\Phi}^\circ]\right)}{d\mathbf{g}}[\mathbf{g}'] \right\rangle_{\Re} \\
=&\left\langle \Phi^\circ, \frac{d\left(\left(\frac{\partial L}{\partial \mathbf{\Phi}}\right)^*[1]\right)}{d\mathbf{g}}[\mathbf{g}'] \right\rangle_P \\
=&\left\langle \Phi^\circ, \frac{\partial\left(\left(\frac{\partial L}{\partial \mathbf{\Phi}}\right)^*[1]\right)}{\partial \mathbf{\Phi}}[\mathbf{\Phi}'] + \frac{\partial\left(\left(\frac{\partial L}{\partial \mathbf{\Phi}}\right)^*[1]\right)}{\partial \mathbf{g}}[\mathbf{g}'] + \frac{\partial\left(\left(\frac{\partial L}{\partial \mathbf{\Phi}}\right)^*[1]\right)}{\partial \xi}[\xi'] \right\rangle_P \\
=&\left\langle \mathbf{\Phi}^\circ, \frac{\partial\left(\left(\frac{\partial L}{\partial \mathbf{\Phi}}\right)^*[1]\right)}{\partial \mathbf{\Phi}}[\mathbf{\Phi}'] + \frac{\partial\left(\left(\frac{\partial L}{\partial \mathbf{\Phi}}\right)^*[1]\right)}{\partial \mathbf{g}}[\mathbf{g}'] - \left\langle \xi', \left(\frac{\partial F}{\partial \mathbf{\Phi}}\right)^*[1] \right\rangle_V \right\rangle_P \\
=&0.
\end{aligned} \tag{3.31}$$

As the variation $\mathbf{\Phi}^\circ$ is arbitrary the term inside the inner product has to be zero and thus gives an equation for the determination of ξ'.

To summarize, given some control variation $\mathbf{g}'$ the product of the Hessian matrix with this control variation can be obtained in the following way:

1. compute the state variables $\mathbf{\Phi}$ from the constraints $\mathbf{F} = 0$
2. compute the Lagrangian multipliers ξ from the adjoint equations (3.10)
3. compute the variation of the state variables from the sensitivity equations (3.5)
4. compute the variation of the Lagrangian multipliers from equation (3.31)
5. compute $\frac{d^2\Im}{d\mathbf{g}^2}[\cdot, \mathbf{g}']$ from equation (3.30)

Note, that if only $\mathbf{g}'$ changes and $\mathbf{g}$ stays the same only the last three steps have to be recomputed.

[2]This procedure is analogous to the procedure in section 3.1 for obtaining the sensitivity equations.

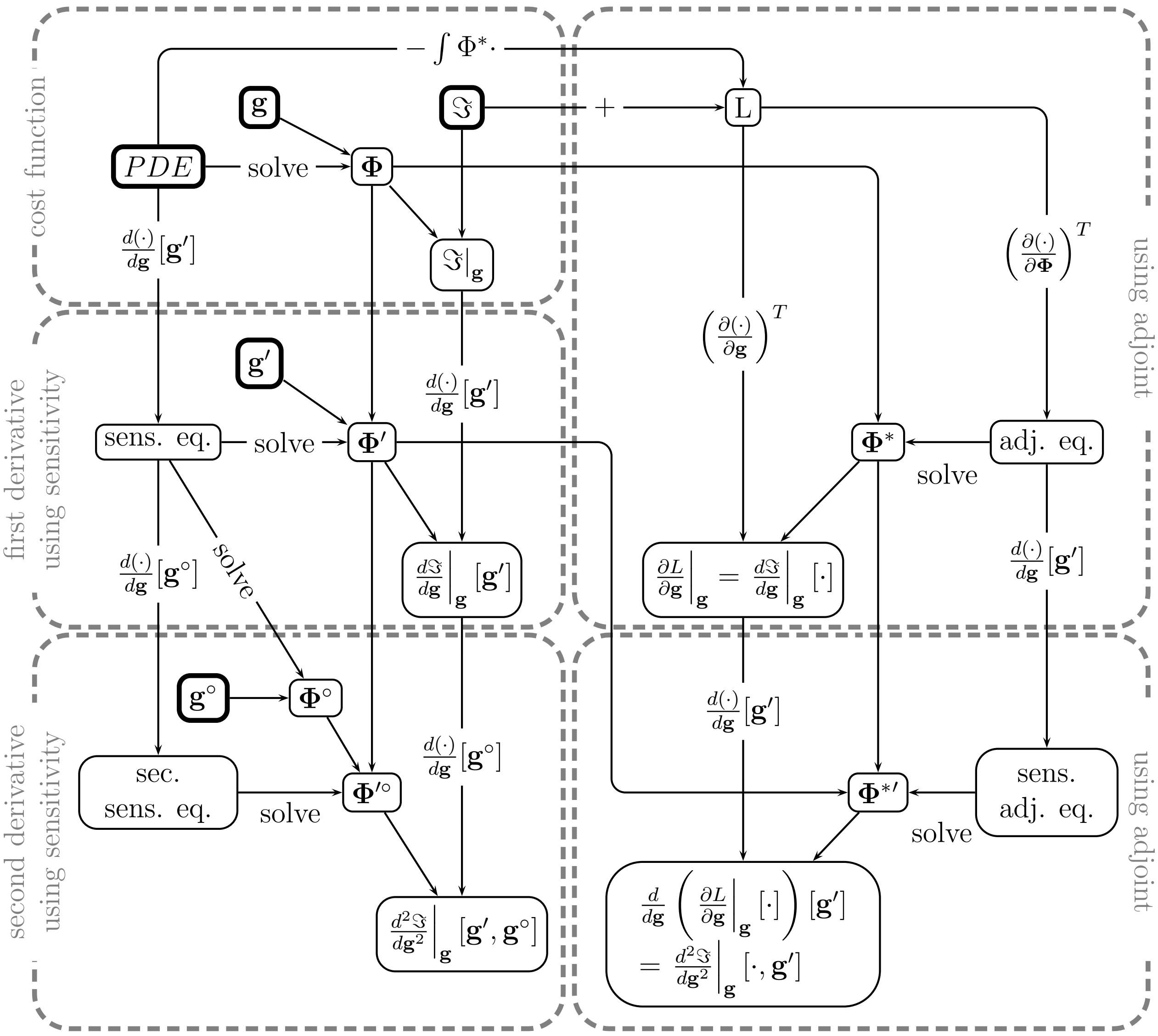

cost function
first derivative using sensitivity
second derivative using sensitivity
first derivative using adjoint
second derivative using adjoint
$-\int \Phi^* \cdot$
$\mathbf{g}$
$\Im$
$+$
L
PDE
solve
$\boldsymbol{\Phi}$
$\Im|_{\mathbf{g}}$
$\frac{d(\cdot)}{d\mathbf{g}}[\mathbf{g}']$
$\left(\frac{\partial(\cdot)}{\partial\mathbf{g}}\right)^T$
$\left(\frac{\partial(\cdot)}{\partial\boldsymbol{\Phi}}\right)^T$
$\mathbf{g}'$
sens. eq.
$\boldsymbol{\Phi}'$
$\boldsymbol{\Phi}^*$
adj. eq.
$\frac{d(\cdot)}{d\mathbf{g}}[\mathbf{g}^\circ]$
$\left.\frac{d\Im}{d\mathbf{g}}\right|_{\mathbf{g}}[\mathbf{g}']$
$\left.\frac{\partial L}{\partial \mathbf{g}}\right|_{\mathbf{g}} = \left.\frac{d\Im}{d\mathbf{g}}\right|_{\mathbf{g}}[\cdot]$
$\mathbf{g}^\circ$
$\boldsymbol{\Phi}^\circ$
sec. sens. eq.
$\boldsymbol{\Phi}'^\circ$
$\boldsymbol{\Phi}^{*\prime}$
sens. adj. eq.
$\left.\frac{d^2\Im}{d\mathbf{g}^2}\right|_{\mathbf{g}}[\mathbf{g}',\mathbf{g}^\circ]$
$\frac{d}{d\mathbf{g}}\left(\left.\frac{\partial L}{\partial \mathbf{g}}\right|_{\mathbf{g}}[\cdot]\right)[\mathbf{g}'] = \left.\frac{d^2\Im}{d\mathbf{g}^2}\right|_{\mathbf{g}}[\cdot,\mathbf{g}']$

Figure 3.1: Illustration of the procedure to obtain zero, first and second derivative information using either the sensitivity or adjoint approach. The thickly framed quantities, namely the PDE, cost functional $\Im$ and control $\mathbf{g}$ together with their variations $\mathbf{g}', \mathbf{g}^\circ$ must be given. $\cdot'$ and $\cdot^\circ$ indicates variations with respect to $\mathbf{g}'$ and $\mathbf{g}^\circ$ and $\cdot^*$ are adjoint variables. The arrows indicate the information necessary to compute/calculate the particular quantities. Arrows marked with "solve“ constitude the computationally expensive parts of the procedure that require a solution of differential equations.

The above procedure introduces a way to calculate a Hessian×vector product. To obtain the full Hessian one could compute this product for N different control variations $\mathbf{g}'$, where N is the dimension of the control space. However, such an approach is not feasible for high N, for the same reasons it wasn't feasible to compute the gradient through N solutions of the sensitivity equation.

$$\begin{tikzcd}
\boxed{\text{non-linear PDE}} \arrow[r, dashed, "\text{discretize}"] \arrow[d, "\text{linearise}"] & \boxed{\text{non-linear discrete PDE}} \arrow[d, dashed, "\text{linearise}"] \\
\boxed{\text{linear PDE}} \arrow[d, "\text{transpose}"] & \boxed{\text{linear discrete PDE}} \arrow[d, dashed, "\text{transpose}"] \\
\boxed{\text{linear adjoint PDE}} \arrow[r, "\text{discretize}"] & \boxed{\text{discrete adjoint PDE}}
\end{tikzcd}$$

Figure 3.2: Illustration of the ways for obtaining the discretized adjoint equations of the partial differential equations. Note, that the two ways are not equivalent.

The procedure for obtaining derivative information of a cost functional under implicit constraints is illustrated in a flowchart in figure 3.1.

3.3. Continuous and Discrete Adjoint

The discretized adjoint equations of a PDE can be obtained in two different ways. The so called continuous adjoint is obtained by calculating the adjoint equations of the PDE and then discretizing these equations. The discrete adjoint is obtained by discretizing the PDE and then calculating the adjoint of this discrete system. The two ways are illustrated in figure 3.2. The ways for obtaining the discrete and the continuous adjoint do not commute and the results may differ.

The continuous approach has the advantage that, given the adjoint equations of the PDE, the discretization can be done using the same numerical schemes used for the discretization of the PDE. Furthermore, the separate discretization of the flow and adjoint equations makes the continuous adjoint approach more flexible, e.g. the flow and adjoint equations could be solved using different grids. On the other hand, this flexibility implies ambiguities and the optimal choice for the discretization of the continuous adjoint is often not clear. Moreover, these ambiguities lead to inconsistencies between the discretized PDE and the discretized continuous adjoint equations. Consequently, errors are introduced in the adjoint solution. As the discrete approach gives the adjoint of the discretized system it is thus exact in the discretized framework. On the other hand the implementation of the discrete adjoint can become cumbersome.

The implementation of the discrete and continuous adjoint approach are explained in detail in section 4.3 and 4.6.3, respectively. The differences between these two approaches constitute a major topic of this work.

4. Numerics

4.1. Spatial Discretization

In this work FD [56, 94, 99] are used for the spatial discretization. Given a linear operator $\hat{L}$ (explicit choices for the linear operator will be discussed in the following sections) and some function $f(x) : \Re \to \Re$ the basic idea is to express the value $\hat{L}f(x_0)$ at some position x_0 as a sum of function values at N nearby positions x_i

$$\hat{L}f_0 \approx \sum_{i=1}^{N} a_i f_i, \tag{4.1}$$

where the definition $f_i = f(x_i)$ holds and a_i are scalar factors chosen such that the behavior of the linear operator is approximated as best as possible.

In this work two ways of defining "best as possible" are explained and explored further. The first way stems from a Taylor series expansion and results in

$$f_i = \sum_{n=0}^{M-1} \frac{(x_i - x_0)^n}{n!} f_0^{(n)} + O\left((x_i - x_0)^M\right), \tag{4.2}$$

where $f_0^{(i)}$ indicates the i-th derivative of f at position x_0. Substituting equation (4.2) into equation (4.1) and neglecting terms with orders higher than $M-1$ leads to

$$\hat{L}f_0 \approx \sum_{i=1}^{N} a_i f_i = \sum_{i=1}^{N} a_i \sum_{n=0}^{M-1} \frac{(x_i - x_0)^n}{n!} f_0^{(n)} = \sum_{n=0}^{M-1} f_0^{(n)} \sum_{i=1}^{N} a_i \frac{(x_i - x_0)^n}{n!}. \tag{4.3}$$

Assuming that we can write

$$\hat{L}f_0 = \sum_{n=0}^{\infty} l_n f_0^{(n)} \tag{4.4}$$

one can see from equation (4.3) that the required coefficients a_i must satisfy the following linear system of equations to be accurate up to order $M-1$

$$\sum_{i=1}^{N} \chi_{ni} a_i = l_n \quad \forall\, n = 1 \dots M, \tag{4.5}$$

where

$$\chi_{ni} = \frac{(x_i - x_0)^n}{n!}. \tag{4.6}$$

Note, that this system of equations is closed only for $N = M$.

Another way of investigating the accuracy of a FD approximation is the Von Neumann or Fourier stability analysis, which is based on the Fourier transformation

$$f(x) = \int_{-\infty}^{\infty} \widetilde{f}(k) e^{ikx} dk \tag{4.7}$$
$$\widetilde{f}(k) = \tfrac{1}{2\pi} \int_{-\infty}^{\infty} f(x) e^{-ikx} dk. \tag{4.8}$$

Substituting equation (4.7) into equation (4.1) and exploiting the linearity of the operator $\hat{L}$ leads to

$$\begin{aligned} \hat{L} f_0 &= \hat{L} \int_{-\infty}^{\infty} \widetilde{f}(k) e^{ikx_0} dk = \int_{-\infty}^{\infty} \widetilde{f}(k) \quad \hat{L} \quad e^{ikx_0} dk \\ \approx \sum_{i=1}^{N} a_i f_i &= \sum_{i=1}^{N} a_i \int_{-\infty}^{\infty} \widetilde{f}(k) e^{ikx_i} dk = \int_{-\infty}^{\infty} \widetilde{f}(k) \left(\sum_{i=1}^{N} a_i e^{ik(x_i - x_0)} \right) e^{ikx_0} dk. \end{aligned} \tag{4.9}$$

Equation (4.9) gives rise to the definition of the transfer functions for the linear operator $\hat{L}$ and its FD approximation

$$\widetilde{L}(k) = e^{-ikx_0} \hat{L} e^{ikx_0} \tag{4.10}$$
$$\widetilde{L}_a(k) = \textstyle\sum_{i=1}^{N} a_i e^{ik(x_i - x_0)}. \tag{4.11}$$

The transfer function or dispersion relation gives the amplification behavior of a linear operator in dependence of wave number. By comparing the expressions in equation (4.9) it becomes obvious that the coefficients a_i must be chosen to satisfy

$$\widetilde{L}(k) \approx \widetilde{L}_a(k) \tag{4.12}$$

to give an adequate approximation to $\hat{L}$. Schemes optimized regarding equation (4.12) are referred to as dispersion relation preserving (DRP) schemes [56, 94].

For concreteness we choose equation (4.12) to be fulfilled in a least square sense

$$\begin{aligned}\int_0^\infty \Bigg\{ g_r(k) \left[Re(\widetilde{L}(k)) - \sum_{i=1}^N a_i \cos(k(x_i - x_0)) \right]^2 \\ + g_i(k) \left[Im(\widetilde{L}(k)) - \sum_{i=1}^N a_i \sin(k(x_i - x_0)) \right]^2 \Bigg\} dk \overset{!}{=} \min,\end{aligned} \tag{4.13}$$

where some arbitrary weighting functions $g_r(k)$ and $g_i(k)$ were introduced and $Re(x)$ and $Im(x)$ represent the real and imaginary part of x, respectively. Expanding equation (4.13) and neglecting terms constant with respect to the coefficients a_i as they do not affect the resulting optimal coefficients leads to the equivalent condition

$$\begin{aligned} I \equiv \int_0^\infty \Bigg\{ & g_r(k) \Bigg[\sum_{i,j=1}^N a_i a_j \cos(k(x_i - x_0)) \cos(k(x_j - x_0)) \\ & - 2Re(\widetilde{L}(k)) \sum_{i=1}^N a_i \cos(k(x_i - x_0)) \Bigg] + \\ & g_i(k) \Bigg[\sum_{i,j=1}^N a_i a_j \sin(k(x_i - x_0)) \sin(k(x_j - x_0)) \\ & - 2Im(\widetilde{L}(k)) \sum_{i=1}^N a_i \sin(k(x_i - x_0)) \Bigg] \Bigg\} dk \\ = & \sum_{i,j=1}^N a_i a_j \int_0^\infty [g_r(k) \cos(k(x_i - x_0)) \cos(k(x_j - x_0)) \\ & + g_i(k) \sin(k(x_i - x_0)) \sin(k(x_j - x_0))] \, dk \\ & - \sum_{i=1}^N 2a_i \int_0^\infty \Big[g_r(k) Re(\widetilde{L}(k)) \cos(k(x_i - x_0)) \\ & + g_i(k) Im(\widetilde{L}(k)) \sin(k(x_i - x_0)) \Big] \, dk \\ = & \sum_{i,j=1}^N a_i \Omega_{ij} a_j - \sum_{i=1}^N a_i b_i \overset{!}{=} \min \end{aligned} \tag{4.14}$$

with

$$\begin{aligned} \Omega_{ij} = & \int_0^\infty [g_r(k) \cos(k(x_i - x_0)) \cos(k(x_j - x_0)) \\ & + g_i(k) \sin(k(x_i - x_0)) \sin(k(x_j - x_0))] \, dk \\ b_i = & \quad 2 \int_0^\infty \Big[g_r(k) Re(\widetilde{L}(k)) \cos(k(x_i - x_0)) \end{aligned} \tag{4.15}$$

$$+ g_i(k) Im(\widetilde{L}(k)) \sin(k(x_i - x_0))\Big] \, dk. \tag{4.16}$$

The quadratic minimization problem in equation (4.14) can be solved by setting the derivative with respect to the coefficients a_i to zero, yielding a linear system of equations.

However, before doing so let's first combine the Taylor and the Fourier approach for obtaining a_i. The idea is to fulfill the Taylor condition (4.5) up to some order $M < N$ and using the remaining $N - M$ coefficients to optimize the transfer function.

This leads to a constraint optimization problem. Thus we define a Lagrangian (see section 3.1) as

$$L = \sum_{i,j=1}^{N} a_i \Omega_{ij} a_j - \sum_{i=1}^{N} a_i b_i + \sum_{i=1}^{M} \xi_i \left(\sum_{j=1}^{N} \chi_{ij} a_j - l_i \right), \tag{4.17}$$

where ξ_i are Lagrangian multipliers. Differentiation with respect to the coefficient a_i and the Lagrangian multipliers ξ_i gives the optimality system

$$\frac{\partial L}{\partial a_k} = 2 \sum_{i=1}^{N} \Omega_{ki} a_i - b_k + \sum_{i=1}^{M} \xi_i \chi_{ik} \overset{!}{=} 0 \qquad \forall\ k = 1 \dots N \tag{4.18}$$

$$\frac{\partial L}{\partial \xi_k} = \sum_{i=1}^{N} \chi_{ki} a_i - l_k \overset{!}{=} 0 \qquad \forall\ k = 1 \dots M \tag{4.19}$$

where the symmetry of Ω has been used. Equations (4.18) and (4.19) can also be written as

$$\begin{pmatrix} \Omega + \Omega^T & \chi^T \\ \chi & \mathbf{0} \end{pmatrix} \begin{pmatrix} a \\ \xi \end{pmatrix} = \begin{pmatrix} b \\ l \end{pmatrix} \tag{4.20}$$

which lightens the fact that the FD coefficients a_i can be obtained by the solution of a linear system of equations.

4.1.1. Derivatives

To obtain a FD scheme the linear Operator is now chosen to be the n-th derivative

$$\hat{L} = \frac{\partial^n}{\partial x^n}. \tag{4.21}$$

Furthermore we choose

$$g_r(k) = \Theta\left(k_{r,0} - k\right) \tag{4.22}$$

$$g_i(k) = \Theta\left(k_{i,0} - k\right), \tag{4.23}$$

where Θ is the Heaviside step function. The idea behind these choices is to optimize the coefficients for low wavelength only and ignore higher wavelength were FD are unlikely to give valid results anyway.

From equation (4.4), equation (4.10), equation (4.15) and equation (4.16) it follows that

$$
\begin{aligned}
l_i &= \delta_{i,n} && (4.24)\\
\widetilde{L}(k) &= (ik)^n && (4.25)\\
\Omega_{ij} &= \frac{\sin(k_{r,0}(x_i+x_j-2x_0)) - \sin(k_{i,0}(x_i+x_j-2x_0))}{2\,(x_i+x_j-2x_0)}\\
&\quad + \frac{\sin(k_{r,0}(x_i-x_j)) + \sin(k_{i,0}(x_i-x_j))}{2\,(x_i-x_j)} && (4.26)\\
b_i &= -\frac{2k_{i,0}}{x_i-x_0}\cos(k_{r,0}(x_i-x_0))\frac{2}{(x_i-x_0)^2}\sin(k_{i,0}(x_i-x_0)). && (4.27)
\end{aligned}
$$

Table 4.1 lists the coefficients for the first derivative used in this work obtained with the choices $x_0 = 0$, $\mathbf{x} = (-4,-3,-2,-1,0,1,2,3,4)^T$ and $k_{r,0} = k_{i,0} = \frac{\pi}{2}$.

In principle, the same methodology could be applied to obtain high order FD schemes near the boundary. However, non central FD schemes obtained with the methodology described above tend to become unstable. Thus, the boundary and inner coefficients are chosen such that the first derivative operator fulfills the so called summation by parts (SBP) property [91]. The SBP property is a formal numeric representation of the integration by parts rule and leads to a stable boundary closure for the first derivative operator. To be able to fulfill the additional SBP rule without increasing the stencil the order of the schemes has to be decreased at the boundaries, thus, accuracy is sacrificed for numerical stability at the boundaries. The boundary coefficients used in this work are listed in table 4.1 and can also be found in [48]. The application of boundary conditions change the effective FD operator. Thus, boundary conditions can flaw the SBP property, eventually leading to a unstable FD operator for the overall numerical scheme [18]. This observation leads to the development of simultaneous approximation term (SAT) schemes, where the boundary conditions (BC) are not applied directly, but imposed implicitly through a forcing term [18]. This forcing has to be chosen weak enough to obtain a stable numerical scheme. On the other hand it has to be strong enough to fulfill the BC within an desired precision. A similar approach is realized for the characteristic boundary conditions (CBC) introduced in section 4.7.1. However, no proof of stability for the SBP scheme described above in conjunction with the CBC will be given in this work. Nonetheless, numerical experiments indicate stability of the underlying numerical system, at least for the numerical and physical setup in this work.

4.1.2. Filtering

The FD scheme used in this work is unstable due to aliasing errors. Thus, the solutions have to be regularized from time to time with a filter which reduces small scales (low pass

central coefficients	
$\alpha_{-4} = 0.0059398042783167$	$\alpha_1 = 0.8331572598964345$
$\alpha_{-3} = -0.0523054923365671$	$\alpha_2 = -0.2331572598964345$
$\alpha_{-2} = 0.2331572598964345$	$\alpha_3 = 0.0523054923365671$
$\alpha_{-1} = -0.8331572598964345$	$\alpha_4 = -0.0059398042783167$
$\alpha_0 = 0.0$	
boundary coefficients	
$\alpha_{00} = -1.585514266533103$	$\alpha_{01} = 2.008723732799078$
$\alpha_{02} = -0.01308559919861748$	$\alpha_{03} = -0.65794293386758834$
$\alpha_{04} = 0.24781906680023041$	$\alpha_{05} = 0$
$\alpha_{10} = -0.4546274514421262$	$\alpha_{11} = 0$
$\alpha_{12} = 0.21294118108792919$	$\alpha_{13} = 0.40745097115747494$
$\alpha_{14} = -0.18058822836810620$	$\alpha_{15} = 0.01482352756482832$
$\alpha_{20} = 0.00663862172240287$	$\alpha_{21} = -0.4773187801117445$
$\alpha_{22} = 0$	$\alpha_{23} = 0.30708203333605406$
$\alpha_{24} = 0.25236237527323959$	$\alpha_{25} = -0.07920868033033617$
$\alpha_{26} = -0.00955556988961586$	$\alpha_{27} = 0$
$\alpha_{30} = 0.1664613995014180$	$\alpha_{31} = -0.4554757153982624$
$\alpha_{32} = -0.15314238751322896$	$\alpha_{33} = 0$
$\alpha_{34} = 0.48145586721457543$	$\alpha_{35} = -0.07649738506727780$
$\alpha_{36} = 0.04196360193662286$	$\alpha_{37} = -0.00476538067384707$
$\alpha_{40} = -0.08600120227144731$	$\alpha_{41} = 0.2769005367876363$
$\alpha_{42} = -0.17262754851385544$	$\alpha_{43} = -0.66039073916329220$
$\alpha_{44} = 0$	$\alpha_{45} = 0.84767356821082549$
$\alpha_{46} = -0.25657769423135097$	$\alpha_{47} = 0.05755953138801333$
$\alpha_{48} = -0.00653645220651922$	$\alpha_{49} = 0$
$\alpha_{50} = 0$	$\alpha_{51} = -0.02035922973188162$
$\alpha_{52} = 0.04853261495732707$	$\alpha_{53} = 0.09398672096440634$
$\alpha_{54} = -0.75928366872272205$	$\alpha_{55} = 0$
$\alpha_{56} = 0.82124427608462746$	$\alpha_{57} = -0.22982343710396821$
$\alpha_{58} = 0.05155759693498167$	$\alpha_{59} = -0.00585487338277167$

Table 4.1: Finite difference coefficients for computation of first derivative. For the central coefficients the index in α_i indicates the difference to the grid point where the derivative is to be calculated. For the boundary coefficients the indices in α_{ij} indicate the coefficient for point j for the derivative at point i.

filter).

In this work such filters are constructed by interpolating from adjacent grid points to the grid point to be filtered with the highest possible order. This interpolated value is averaged with the grid points value such that even odd waves are removed completely. Even odd

central coefficients	
$\alpha_{-5} = 1/1024$	$\alpha_1 = 210/1024$
$\alpha_{-4} = -10/1024$	$\alpha_2 = -120/1024$
$\alpha_{-3} = 45/1024$	$\alpha_3 = 45/1024$
$\alpha_{-2} = -120/1024$	$\alpha_4 = -10/1024$
$\alpha_{-1} = 210/1024$	$\alpha_5 = 1/1024$
$\alpha_0 = 772/1024$	
boundary coefficients	
$\alpha_{00} = 15/16$	$\alpha_{01} = 4/16$
$\alpha_{02} = -6/16$	$\alpha_{03} = 4/16$
$\alpha_{04} = -1/16$	$\alpha_{05} = 0$
$\alpha_{10} = 1/16$	$\alpha_{11} = 12/16$
$\alpha_{12} = 6/16$	$\alpha_{13} = -4/16$
$\alpha_{14} = 1/16$	$\alpha_{15} = 0$
$\alpha_{20} = -1/16$	$\alpha_{21} = 4/16$
$\alpha_{22} = 10/16$	$\alpha_{23} = 4/16$
$\alpha_{24} = -1/16$	$\alpha_{25} = 0$
$\alpha_{30} = 1/64$	$\alpha_{31} = -6/64$
$\alpha_{32} = 15/64$	$\alpha_{33} = 44/64$
$\alpha_{34} = 15/64$	$\alpha_{35} = -6/64$
$\alpha_{36} = 1/64$	$\alpha_{37} = 0$

(a) 10^{th} order filter

central coefficients	
$\alpha_{-3} = 1/64$	$\alpha_1 = 15/64$
$\alpha_{-2} = -6/64$	$\alpha_2 = -6/64$
$\alpha_{-1} = 15/64$	$\alpha_3 = 1/64$
$\alpha_0 = 44/64$	
boundary coefficients	
$\alpha_{00} = 15/16$	$\alpha_{01} = 4/16$
$\alpha_{02} = -6/16$	$\alpha_{03} = 4/16$
$\alpha_{04} = -1/16$	$\alpha_{05} = 0$
$\alpha_{10} = 1/16$	$\alpha_{11} = 12/16$
$\alpha_{12} = 6/16$	$\alpha_{13} = -4/16$
$\alpha_{14} = 1/16$	$\alpha_{15} = 0$
$\alpha_{20} = -1/16$	$\alpha_{21} = 4/16$
$\alpha_{22} = 10/16$	$\alpha_{23} = 4/16$
$\alpha_{24} = -1/16$	$\alpha_{25} = 0$

(b) 6^{th} order filter

central coefficients	
$\alpha_{-2} = -1/16$	$\alpha_1 = 4/16$
$\alpha_{-1} = 4/16$	$\alpha_2 = 1/16$
$\alpha_0 = 10/16$	
boundary coefficients	
$\alpha_{00} = 1/2$	$\alpha_{01} = 1/2$
$\alpha_{10} = 1/4$	$\alpha_{11} = 2/4$
$\alpha_{12} = 1/4$	$\alpha_{13} = 0$

(c) 4^{th} order filter

Table 4.2: Finite difference coefficients for (a) 10^{th}-order, (b) 6^{th}-order and (c) 4^{th}-order accurate filter. The meaning of the indices is the same as in table 4.1.

waves are waves where every grid point has the same amplitude but different sign than its neighboring grid points. The coefficients for filters of different orders together with their boundary coefficients are listed in table 4.2. The filter coefficients for the boundary closure can also be found in [56].

1^{th} step

α	β	$\frac{t}{\Delta t}$
0	0.2687454388871343849	0
−0.60512264332862261228	0.8014706973220802933	0.26874543888713438496
−2.04375640234761394333	0.5051570426942272253	0.58522806929524303469
−0.74069990637544192841	0.5623568037900029640	0.68270664478424678821
−4.42317651302968168941	0.05900655127758823335	1.1646854837729261436

2^{nd} step

α	β	$\frac{t}{\Delta t}$
0	0.1158488818128556168	0
−0.4412737715387738256	0.3728769905165286498	0.11584888181285561688
−1.073982008079781868	0.7379536892143529568	0.32418503640412806853
−1.706357079125675880	0.5798110936631103958	0.61932082035177792368
−2.797929316268244305	1.031284991300145194	0.80344726663359079059
−4.091353712091916045	0.15	0.91841664452065965078

Table 4.3: Coefficients for the RK scheme for time integration used in this work together with the time of substeps.

4.2. Temporal Discretization

In this work a two step explicit RK scheme is used for time integration. The substeps of this scheme are given by

$$\mathbf{k}_{i+1} = \alpha_i \mathbf{k}_i + \Delta t \, \mathbf{R}\left(\boldsymbol{\Phi}_i\right) \qquad i \in \{1 \ldots a_j\} \tag{4.28}$$

$$\boldsymbol{\Phi}_{i+1} = \boldsymbol{\Phi}_i + \beta_i \mathbf{k}_{i+1} \qquad i \in \{1 \ldots a_j\}, \tag{4.29}$$

where $a_1 = 5$ and $a_2 = 6$ for the first and second RK step, respectively. u_i are the flow variables at the i-th substep and $\mathbf{R}(u_i)$ is the right hand side (RHS) of the Navier Stokes equations. The coefficients α_i and β_i can be reviewed in [86, 7] and are given together with the time position of the substeps in table 4.3. Note, that $\alpha_1 = 0$ for both RK steps and thus the scheme in (4.28)-(4.29) is explicit.

Both RK steps have fourth order accuracy and the coefficients are optimized in Fourier space leading to a so called low-dissipation low-dispersion Runge Kutta (LDDRK) scheme. The amplitude and phase related errors for this RK scheme are shown in figure 4.1. See [45] for further details and the definitions of the dissipation ratio and phase error.

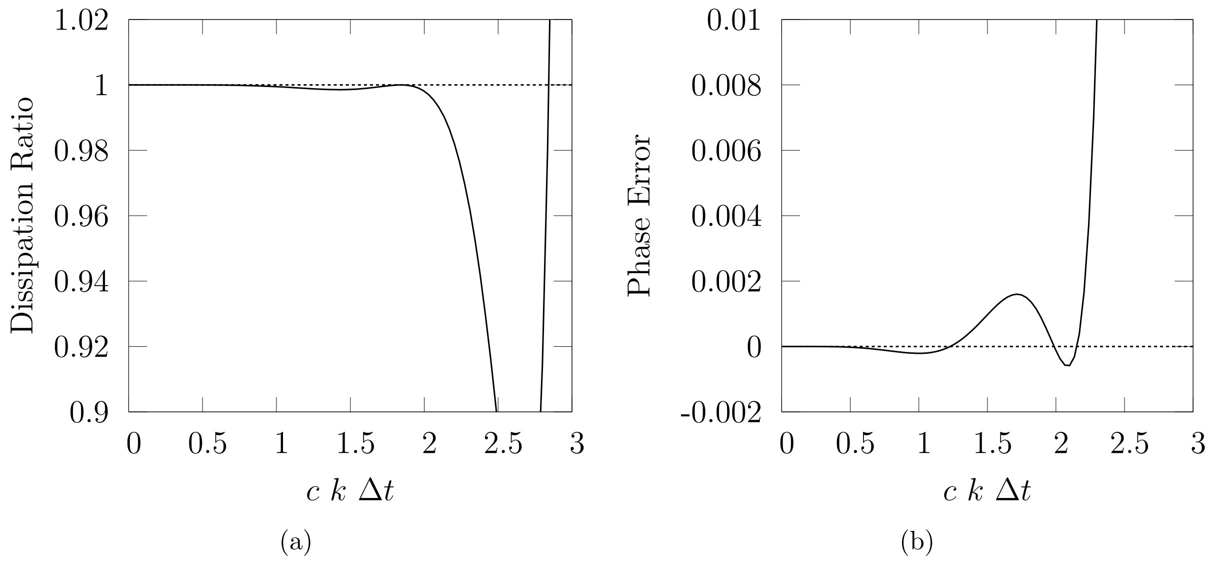


Figure 4.1: (a) Dissipation ratio and (b) phase error for the RK-coefficients shown in table 4.3. The dotted lines represent the exact solution.

4.3. Implementation of Discrete Adjoint

In order to calculate the adjoint of a discrete system AD tools [39, 6] can be used. These AD tools are either manipulating the source code in a precompiler step or keep track of the computations and obtain derivative information by applying chain/product rule to each computation.

However, as within this work a code was developed that derives the adjoint system of a given system of equations it was more practical to use this program to implement the discrete adjoint. Nevertheless, some parts had to be implemented ”by hand“ and the details of this implementation are described in the following.

4.3.1. Discrete Adjoint of a Runge-Kutta Step

The complete series of N Runge-Kutta substeps together with filtering and control can be written as [45]

$$\mathbf{k}_0 = 0 \tag{4.30}$$

$$\mathbf{\Phi}_0 = \mathbf{\Phi}_{\text{init}} \tag{4.31}$$

$$\mathbf{k}_s = \alpha_{s-1}\mathbf{k}_{s-1} + \Delta t \left[\mathbf{R}\left(\mathbf{\Phi}_{s-1}\right) + \sum_{i=0}^{M} \gamma_{s-1,i} g_i \right] \qquad s \in \{1 \ldots N\} \tag{4.32}$$

$$\mathbf{\Phi}_s = F_s\left[\mathbf{\Phi}_{s-1} + \beta_{s-1}\mathbf{k}_s\right] \qquad s \in \{1 \dots N\},, \tag{4.33}$$

where F_s is the discrete representation of a filter operator at step s, g_i are $M+1$ control vectors and $\gamma_{s,i}$ are scalars giving the strength of control g_i at step s. The rest of the notation is equal to equation (4.28)-(4.29).

Expressing the cost functional as a sum over the contributions of the individual time steps $\Im = \sum_{s=0}^{N} \Im_s(\mathbf{\Phi}_s)$ the Lagrangian is given by

$$\begin{aligned} L =& \sum_{s=0}^{N} \Im_s(\mathbf{\Phi}_s) - \sum_{s=1}^{N} \xi_s^T \left[\mathbf{k}_s - \alpha_{s-1}\mathbf{k}_{s-1} - \Delta t \left[\mathbf{R}\left(\mathbf{\Phi}_{s-1}\right) + \sum_{i=0}^{M} \gamma_{s-1,i} g_i\right]\right] \\ & - \sum_{s=1}^{N} \omega_s^T \left[\mathbf{\Phi}_s - F_s\left[\mathbf{\Phi}_{s-1} + \beta_{s-1}\mathbf{k}_s\right]\right] - \xi_0^T \mathbf{k}_0 - \omega_0^T\left[\mathbf{\Phi}_0 - \mathbf{\Phi}_{\text{init}}\right], \end{aligned} \tag{4.34}$$

where ξ and ω are Lagrangian multipliers used to add the Runge-Kutta steps as a constraint. A variation with respect to the state variables $\mathbf{\Phi}_s$ and ω_s gives

$$\begin{aligned} \frac{\partial L}{\partial \mathbf{\Phi}}\mathbf{\Phi}'_s =& \sum_{s=0}^{N} \left.\frac{\partial \Im_s}{\partial \mathbf{\Phi}}\right|_{\mathbf{\Phi}_s} \mathbf{\Phi}'_s - \sum_{s=1}^{N} \xi_s^T \left[-\Delta t \left.\frac{\partial \mathbf{R}}{\partial \mathbf{\Phi}}\right|_{\mathbf{\Phi_{s-1}}} \mathbf{\Phi}'_{s-1}\right] \\ & - \sum_{s=1}^{N} \omega_s^T \left[\mathbf{\Phi}'_s - F_s \mathbf{\Phi}'_{s-1}\right] - \omega_0^T \mathbf{\Phi}'_0 \end{aligned} \tag{4.35}$$

$$\frac{\partial L}{\partial \mathbf{k}}\mathbf{k}'_s = - \sum_{s=1}^{N} \xi_s^T \left[\mathbf{k}'_s - \alpha_{s-1}\mathbf{k}'_{s-1}\right] - \sum_{s=1}^{N} \omega_s^T \left[-F_s \beta_{s-1}\mathbf{k}'_s\right] - \xi_0^T \mathbf{k}'_0. \tag{4.36}$$

Note, that the notation $\frac{\partial L}{\partial \mathbf{\Phi}}\left[\mathbf{\Phi}'_s\right]$, introduced in chapter 3.1, is substituted with $\frac{\partial L}{\partial \mathbf{\Phi}}\mathbf{\Phi}'_s$ to illuminate the fact that in the discrete case the directional derivative can be expressed as a matrix vector multiplication. The equations above lead to the adjoint RK integration

$$\omega_N = \left(\left.\frac{\partial \Im_N}{\partial \mathbf{\Phi}}\right|_{\mathbf{\Phi}_N}\right)^T \tag{4.37}$$

$$\xi_N = \beta_{N-1} F_N^T \omega_N \tag{4.38}$$

$$\omega_s = F_{s+1}^T \omega_{s+1} + \Delta t \left(\left.\frac{\partial \mathbf{R}}{\partial \mathbf{\Phi}}\right|_{\mathbf{\Phi_s}}\right)^T \xi_{s+1} + \left(\left.\frac{\partial \Im_s}{\partial \mathbf{\Phi}}\right|_{\mathbf{\Phi}_s}\right)^T \qquad s \in \{0 \dots N-1\} \tag{4.39}$$

$$\xi_s = \alpha_s \xi_{s+1} + \beta_{s-1} F_s^T \omega_s \qquad s \in \{1 \dots N-1\} \tag{4.40}$$

$$\xi_0 = \alpha_0 \xi_1. \tag{4.41}$$

Note, that in analogy to the continuous example in section 3.1.2 a boundary condition is given at the final iteration N. This implies that the system of equations (4.37)-(4.41) has to be solved backwards starting at iteration N.

```
k = 0
for $m = 1,\dots,N$
    $k_i += D_{im}x_m$
end
```
(a)

```
k = 0
for $m = 1,\dots,N$
    $k_m += D_{im}x_i$
end
```
(b)

Algorithm 7: Algorithms for obtaining (a) $\mathbf{k} = D\mathbf{x}$ and (b) $\mathbf{k} = D^T\mathbf{x}$, where D is a matrix with non zero entries in the i^{th} row, only.

The gradient of the cost functional is given by

$$\left(\frac{d\Im}{dg_i}\right)^T = \left(\frac{\partial L}{\partial g_i}\right)^T = \Delta t \sum_{s=1}^{N} \gamma_{s-1,i}\xi_s \quad i \in \{0\dots M\} \tag{4.42}$$

4.3.2. Discrete Adjoint of Right Hand Side

The adjoint RK integration in equations (4.37)-(4.41) reveals that the transpose of the linearized and discretized Navier-Stokes operator $\left(\frac{\partial R}{\partial \mathbf{\Phi}}\big|_{\mathbf{\Phi_s}}\right)^T$ is needed. As the operator $\frac{\partial R}{\partial \mathbf{\Phi}}\big|_{\mathbf{\Phi_s}}$ is time dependent computing the single components of this operator would not be efficient and consequently only the computation of a matrix vector product is implemented. However, without the knowledge of these single components special considerations have to be made to be able to compute the transpose of this operator.

Let's start with a discrete linear operator D^i that has non-zero entries in the i^{th} row, only. The vector product $\mathbf{k} = D^i\mathbf{x}$ can be computed with algorithm 7(a). On the other hand the product $\mathbf{k} = D^{i^T}\mathbf{x}$ can be computed by algorithm 7(b). Note, that in both cases only the matrix coefficients of the i^{th} row have to be used. This is the reason for the usefulness of the above algorithm as the coefficients of D^T can be accessed similarly to the way it is done for D. In other words a grid point doesn't need to know the FD coefficients of its surrounding grid points, which eases the implementation. The difference between the two algorithms is also illustrated in figure 4.2. The implementation of the transposed linear operator product is similar to the one described in [27].

In order to compute the product with a general matrix D we state that every matrix can be written as

$$D = \sum_{i=1}^{N} D^i, \tag{4.43}$$

where N is the number of rows of matrix D. In our case only the coefficients for the derivative and filter operators are known. As the Navier Stokes equations consist of several sums and products of derivatives the identity

$$(AB + CD)^T = B^TA^T + D^TC^T \tag{4.44}$$

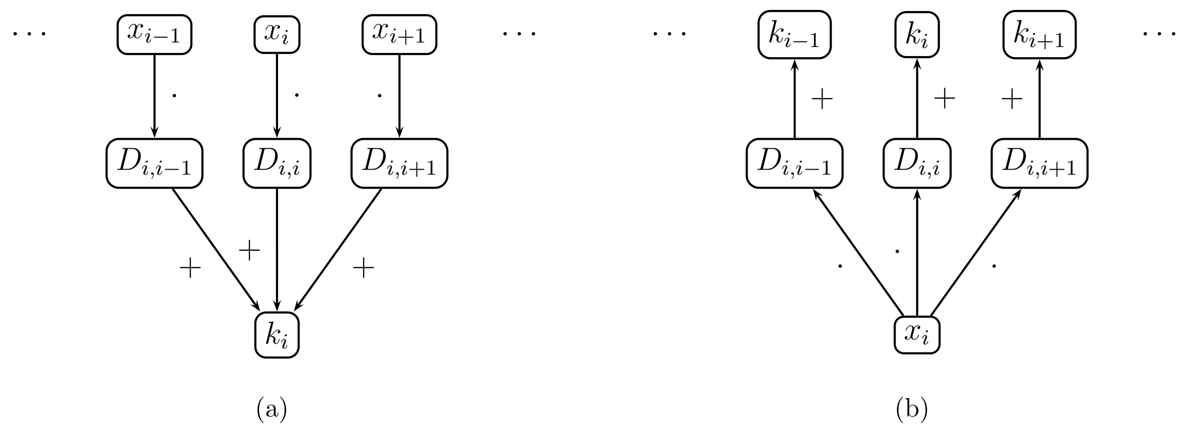


Figure 4.2: Illustration of the differences of (a) algorithm 7(a) and (b) algorithm 7(b). While in the first case several adjacent points are "joint" at some single point this single point is "spread" over several adjacent points in the latter case.

for arbitrary matrixes A, B, C and D has to be used to compute the transpose of the linearized Navier Stokes equations $\left(\frac{\partial R}{\partial \mathbf{\Phi}}\Big|_{\mathbf{\Phi_s}}\right)^T$.

4.4. Automatic Calculation of Adjoint

As the Navier Stokes equations contain many terms, linearizing them and splitting the operators according to equations (4.43) and (4.44) can become a tedious task. Because of this a code was written that, given the PDE, automatically generates code which computes the desired matrix vector product according to the splitting of equations (4.43) and (4.44). The implementation of this code is explained in this section.

For better illustration the general algorithm will be accompanied by the example

$$y(x)\frac{\partial F(y(x)^{-1})}{\partial x} + u(x) = 0 \quad x \in (0,1), \tag{4.45}$$

where $F(\ldots)$ is some arbitrary function, $y(x)$ is a state variable and $u(x)$ is the control. Note, that this system of equations has the trivial solution $y(x) = \text{const}$ if no control is applied. Furthermore, it is not well posed due to the lack of a boundary condition. However, this example is not meant to be physically relevant, but to illustrate the algorithm deriving the adjoint system. Note, that $(A + B)^* = A^* + B^*$ for any operators A and B and thus the adjoint can be derived for every summand in an equation separately. In the following the derivation of the continuous and discrete adjoint will be explained simultaneously, so in the discrete case equation (4.45) has to be fulfilled at a distinct set of grid point and $\frac{\partial}{\partial x}$ represents a discrete operator.

4.4.1. Representation of Equation in Memory

To describe equations a *Variable* is defined. A *Variable* is a C++ class containing information about what kind of variable it represents, which spatial directions it depends on, how it is named in the fortran code that shall be produced as an output etc. The kind of a variable can be a scalar (e.g. $y(x)$), some real value (e.g. 2), the inverse of a scalar or real value (e.g. $y(x)^{-1}$), a differential operator (e.g. $\frac{\partial}{\partial x}[]$), an integral (e.g. $\int_0^1 dx[]$), a function with one argument (e.g. $F[]$) or a closing bracket (]), which marks the end of the argument of a function or differential operator. Here, a new notation was introduced and for the rest of this section, e.g. $\frac{\partial}{\partial x}[y(x)]$ will be written instead of $\frac{\partial y(x)}{\partial x}$ to better illustrate the representation of the equations in memory.

Next, a class *Summand* is defined, which consists of an array of *Variables*. The *Variables* included in this array can be interpreted as being "multiplied" from the left to the right. Recalling the notation introduced above the first part of the left hand side (LHS) in equation (4.45) is represented in memory as an array consisting of 6 *Variables*

$$\underbrace{y(x)}_{1}\underbrace{\frac{\partial}{\partial x}[}_{2}\underbrace{F[}_{3}\underbrace{y(x)^{-1}}_{4}\underbrace{]}_{5}\underbrace{]}_{6}. \tag{4.46}$$

To be able to represent terms consisting of more than one summand a class *Term* is defined which consists of an array of *Summands*. This array is then interpreted as a sum of the different *Summands*. Subtraction is achieved by "multiplying" a *Summand* with -1. It should be noted that this representation of a term is not the most general possible. E.g. the term $\frac{a}{b+c}$ can not adequately be represented. However, such terms are not relevant for the derivation of the adjoint system of the Navier Stokes equations.

With the above considerations the Lagrangian can be evaluated according to equation (3.11). Defining the cost functional for the example case as $\Im = \int_0^1 dx\, y(x)^2$ and introducing the adjoint variable $\widetilde{y}(x)$ the Lagrangian is obtained by "multiplying" equation (4.46) with the adjoint variable and an integral from the left and subsequently adding the cost functional

$$\begin{aligned} L = &\int_0^1 dx[y(x)y(x)] \\ &+ (-1)\int_0^1 dx\left[\widetilde{y}(x)y(x)\frac{\partial}{\partial x}[F[y(x)^{-1}]]\right] + (-1)\int_0^1 dx\,[u(x)]\,. \end{aligned} \tag{4.47}$$

4.4.2. Sensitivities

The next step in the derivation of the adjoint equations is the variation of the Lagrangian with respect to the state variables. This can be achieved by applying the following simple substitution rules

$$\Big(\int_a^b dx[\ldots]\Big)' \rightarrow \int_a^b dx[(\ldots)'] \tag{4.48}$$

$$\Big(\frac{\partial}{\partial x}[\ldots]\Big)' \rightarrow \frac{\partial}{\partial x}[(\ldots)'] \tag{4.49}$$

$$(\text{const } y(x))' \rightarrow \text{const } y'(x) \tag{4.50}$$

$$\big(y(x)^{-1}\big)' \rightarrow (-1)y(x)^{-1}y(x)^{-1}y'(x) \tag{4.51}$$

$$((\ldots)(\ldots))' \rightarrow (\ldots)'(\ldots) + (\ldots)(\ldots)' \tag{4.52}$$

$$F[\ldots] \rightarrow f[\ldots](\ldots)' \tag{4.53}$$

to each appearance of a state variable. $y'(x)$ is a *Variable* representing the variation of a state variable $y(x)$. A user defined function $f[k] = \frac{\partial F[k]}{\partial k}$ must be specified, which represents the partial derivative of function $F[k]$. These rules are just the chain and product rule for differentiation and have to be applied recursively. Note, that these substitution rules work correct no matter if the *Term* represents the continuous or discrete equations. For the example one gets

$$\begin{aligned}\left\langle 1, \frac{\partial L}{\partial y}[y'(x)]\right\rangle_{\Re} = &\int_0^1 dx[2y(x)y'(x)] \\ &+ (-1)\int_0^1 dx\left[\widetilde{y}(x)y'(x)\frac{\partial}{\partial x}[F[y(x)^{-1}]]\right] \\ &+ \int_0^1 dx\left[\widetilde{y}(x)\frac{\partial}{\partial x}[f[y(x)^{-1}]y(x)^{-1}y'(x)]\right].\end{aligned} \tag{4.54}$$

It should be noted, that the above rules substitute $y(x)y(x)$ with $y'(x)y(x) + y(x)y'(x)$. However, after the derivation of the sensitivities an algorithm is applied which identifies and merges identic *Summands* so that finally $2y(x)y'(x)$ is obtained. Furthermore, the code identifies $y(x)y(x)^{-1} = 1$ and $(-1)(-1) = 1$.

4.4.3. Adjoint Equation

To obtain the adjoint equations, equation (4.54) must be transposed/adjoint. Algorithmically this can be achieved by "moving the variation of a state variable to the left" till they next to the integral. If the *Variable* left of the variation variable is a scalar or closing bracket the substitution rule for this "moving" procedure is simple as e.g. $F[\ldots]y'(x) = y'(x)F[\ldots]$. Furthermore, it can be observed that variation variables will by construction always appear outside of functions. Consequently, no "moving rule" is required for e.g. $F[y'(x)]$. In the continuous case the order of a variable and a differentiation operator can be changed by integration by parts, which is realized with the substitution rule

$$\int_a^b dx[\Diamond\frac{\partial}{\partial x}[y'(x)\Box]] \rightarrow (-1)\int_a^b dx[\frac{\partial}{\partial x}[\Diamond]y'(x)\Box] + \big|_a^b dx[\Diamond y'(x)\Box]\,, \tag{4.55}$$

where a new *Variable* $\big|_a^b dx[y(x)] \widehat{=} y(b) - y(a)$ is introduced. In the discrete case the substitution rule

$$\int_a^b dx[\lozenge \frac{\partial}{\partial x}[y'(x)\square]] \to \int_a^b dx[\left(\frac{\partial}{\partial x}\right)^T [\lozenge] y'(x)\square] \tag{4.56}$$

must be applied. Here $\left(\frac{\partial}{\partial x}\right)^T[$ is the transpose of the discrete operator $\frac{\partial}{\partial x}[$. In other words $\frac{\partial}{\partial x}[$ is a formal representation of algorithm 7(a) and $\left(\frac{\partial}{\partial x}\right)^T[$ represents algorithm 7(b). With these substitutions one obtains

$$\begin{aligned}\left\langle y'(x), \left(\frac{\partial L}{\partial y}\right)^* [1]\right\rangle_{\Re} &= \int_0^1 dx[y'(x)2y(x)] \\ &+ (-1)\int_0^1 dx\left[y'(x)\widetilde{y}(x)\frac{\partial}{\partial x}[F[y(x)^{-1}]]\right] \\ &+ (-1)\int_0^1 dx\left[y'(x)\frac{\partial}{\partial x}[\widetilde{y}(x)]f[y(x)^{-1}]y(x)^{-1}\right] \\ &+ \big|_0^1 dx\left[y'(x)\widetilde{y}(x)f[y(x)^{-1}]y(x)^{-1}\right] = 0\end{aligned} \tag{4.57}$$

for the continuous case. From this equation the adjoint equations can be identified as

$$\begin{array}{rll} (-2)y(x) + \widetilde{y}(x)\frac{\partial}{\partial x}[F[y(x)^{-1}]] & & \\ + \quad \frac{\partial}{\partial x}[\widetilde{y}(x)]f[y(x)^{-1}]y(x)^{-1} & = 0 & x \in (0,1) \end{array} \tag{4.58}$$

$$\widetilde{y}(x)f[y(x)^{-1}](-1)y(x)^{-1} = 0 \quad x = 0 \tag{4.59}$$

$$\widetilde{y}(x)f[y(x)^{-1}]y(x)^{-1} = 0 \quad x = 1 \tag{4.60}$$

For the discrete case one obtains

$$\begin{aligned}\left\langle y'(x), \left(\frac{\partial L}{\partial y}\right)^* [1]\right\rangle_{\Re} &= \int_0^1 dx[y'(x)2y(x)] \\ &+ (-1)\int_0^1 dx\left[y'(x)\widetilde{y}(x)\frac{\partial}{\partial x}[F[y(x)^{-1}]]\right] \\ &+ \int_0^1 dx\left[y'(x)\left(\frac{\partial}{\partial x}\right)^T [\widetilde{y}(x)]f[y(x)^{-1}]y(x)^{-1}\right]\end{aligned} \tag{4.61}$$

leading to the discrete adjoint equation

$$\begin{array}{rll} 2y(x) + (-1)\widetilde{y}(x)\frac{\partial}{\partial x}[F[y(x)^{-1}]] & & \\ + \quad \left(\frac{\partial}{\partial x}\right)^T [\widetilde{y}(x)]f[y(x)^{-1}]y(x)^{-1} & = 0 & x \in (0,1). \end{array} \tag{4.62}$$

Note, that the term $\left\langle u'(x), \left(\frac{\partial L}{\partial u}\right)^* [1]\right\rangle_{\Re}$, which is equal to the gradient of the cost function with respect to the control, can be calculated analogously to the procedure described above.

4.5. Sensitivities through Operator Overloading

Recalling a Taylor series expansion in the form $f(x+ih) = \sum_{n=0}^{n=\infty} \frac{(ih)^n}{n!} f^{(n)}(x)$, where i is the imaginary unit, it is easy to see that

$$f(x) = Re(f(x+ih)) + O(h^2) \tag{4.63}$$

$$f^{(1)}(x) = \frac{Im(f(x+ih))}{h} + O(h^3) \tag{4.64}$$

if all derivatives $f^{(n)}(x)$ are real. Thus, the sensitivity of a real function can be computed by substituting all real values by complex values and setting the imaginary part of the input vector equal to the perturbation. This complex differentiation [69] method has two advantages. First, it is rather foolproof and easy to implement as all one has to do is exchanging real by complex values. Second, the derivative is computed without using differences. Because of this the amplitude of the perturbation h is not restricted by the finite machine precision of real numbers. Thus, h can be chosen to be in the order of say 10^{-30}, which means that higher order terms become smaller than machine precision and in this sense the differentiation is exact.

The basic idea of obtaining sensitivities through complex differentiation can be extended to be applicable for higher order sensitivities. First, for some scalar u a vector $\mathbf{u} = (u, u', u^o, u'^o)$ is defined, where $\cdot'$ and $\cdot^o$ indicate variations with respect to two different controls. Next, every operation on the scalar values is substituted by operations on the variation vector such that the components of the resulting vector contain the variations $\cdot'$, $\cdot^o$ and $\cdot'^o$, respectively. In the general case of some function $f(u,v)$ chain rule gives

$$f(\mathbf{u},\mathbf{v}) = \begin{pmatrix} f(u,v) \\ \frac{\partial f}{\partial u}u' + \frac{\partial f}{\partial v}v' \\ \frac{\partial f}{\partial u}u^o + \frac{\partial f}{\partial v}v^o \\ \frac{\partial^2 f}{\partial u^2}u'u^o + \frac{\partial^2 f}{\partial v^2}v'v^o + \frac{\partial^2 f}{\partial u \partial v}(u'v^o + u^o v') + \frac{\partial f}{\partial u}u'^o + \frac{\partial f}{\partial v}v'^o \end{pmatrix}. \tag{4.65}$$

For example, the multiplication becomes

$$\mathbf{u} \cdot \mathbf{v} = \begin{pmatrix} uv \\ u'v + uv' \\ u^o v + uv^o \\ u'^o v + uv'^o + u^o v' + u'v^o \end{pmatrix}. \tag{4.66}$$

Note, that in this case the $\cdot$ doesn't denote a scalar product but some self defined operator the multiplication has to be substituted with. Using operator overloading, which is a feature of most modern programming languages (as e.g. Fortran90 and C++), the substitution rules explained above can be implemented analogously to complex differentiation [101].

However, the methodology just introduced has several advantages. First, the variations don't have to be small, as the higher order terms in equations (4.63)-(4.64) are neglected (the variation is computed directly). This saves some multiplications and book keeping. Secondly, this methodology can, in principle, be extended to variations of arbitrary order in a straight forward manner. Third, it is more efficient than complex differentiation. To see this for example compare the complex multiplication

$$(u + iv')(v + iv') = uv - u'v' + i(uv' + u'v), \tag{4.67}$$

where i is the imaginary unit, with the expression in equation (4.66). It becomes obvious that the multiplication and subtraction of the term $u'v'$ is not performed in the just introduced method.

As already mentioned in section 4.3.2, for the purposes of this thesis a code was written that automatically generates sensitivity equations, making complex differentiation or operator overloading unnecessary. Nevertheless, these two methods were implemented to validate the implementation of the sensitivity equations.

4.6. Governing Equations

4.6.1. Navier Stokes Equations

The governing equations chosen in this work are the compressible Navier-Stokes equations using density, pressure and momentum as flow variables.

$$\begin{aligned}
\frac{\partial \rho}{\partial t} &= -\frac{\partial m_i}{\partial x_i} && (4.68)\\
\frac{\partial m_i}{\partial t} &= -\frac{\partial p}{\partial x_i} - \frac{\partial \rho u_j u_i}{\partial x_j} + \frac{\partial \tau_{ji}}{\partial x_j} && (4.69)\\
\frac{\partial p}{\partial t} &= -\frac{\partial p u_i}{\partial x_i} + \frac{\partial}{\partial x_i}\left(\lambda(\gamma - 1)\frac{\partial T}{\partial x_i}\right) - (\gamma - 1)p\frac{\partial u_i}{\partial x_i} + (\gamma - 1)\tau_{ij}\frac{\partial u_i}{\partial x_j}, && (4.70)
\end{aligned}$$

where ρ is the density, u_i are the velocities in direction i, $\gamma = \frac{C_p}{C_v}$ the ratio of specific heats, T temperature and $\lambda = \frac{\mu C_p}{Pr}$ the heat conductivity with a Prandtl number of $Pr = 0.71$. Furthermore, we have the ideal gas equation

$$p = \rho R T \tag{4.71}$$

and

$$m_i = \rho u_i \tag{4.72}$$

$$\tau_{ij} = \mu s_{ij} \quad = \quad \mu \left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i} - \delta_{ij} \frac{2}{3} \frac{\partial u_k}{\partial x_k} \right), \tag{4.73}$$

where μ is the dynamic viscosity and R the ideal gas constant. For later reference the speed of sound is given by

$$c = \sqrt{\gamma R T} = \sqrt{\gamma \frac{p}{\rho}}. \tag{4.74}$$

In this work the viscosity is chosen to follow Sutherland law

$$\mu \equiv \mu(T) = \mu(T_{ref}) \left(\frac{T}{T_{ref}} \right)^{\frac{3}{2}} \frac{T_{ref} + T_{suth}}{T + T_{suth}}, \tag{4.75}$$

where $T_{suth} = 110.4K$ is the Sutherland temperature and T_{ref} is a reference temperature. The heat capacities C_p and C_v (and hence the gas constant R and ratio of specific heats γ) are kept constant for this work.

For non-dimensionalization we choose a temperature T_{ref}, a density ρ_{ref}, a velocity u_{ref}, and a length l_{ref} as reference parameters. Using these reference values in the appropriate way the equations (4.68)-(4.74) have the same appearance in non-dimensionalized form. Equation (4.75) becomes

$$\mu \equiv \mu(T) = \frac{1}{Re} T^{\frac{3}{2}} \frac{1 + \hat{T}_{suth}}{T + \hat{T}_{suth}} \tag{4.76}$$

in dimensionless form with the dimensionless Sutherland temperature $\hat{T}_{suth} = \frac{T_{suth}}{T_{ref}}$. Here, the dimensionless Reynolds number was introduced as

$$Re = \frac{\rho_{ref} u_{ref} l_{ref}}{\mu}. \tag{4.77}$$

4.6.2. Sensitivity Equations

In order to validate the sensitivities, obtained using the adjoint method, the sensitivity equations have been implemented, too. Variation of the Navier Stokes equations (4.68)-(4.70) gives the sensitivity equations. Calculating the variation of the Navier Stokes equations (4.68)-(4.70) according to equation (3.4) the sensitivity equations read

$$\frac{\partial \rho'}{\partial t} \quad = \quad -\frac{\partial m'_i}{\partial x_i} \tag{4.78}$$

$$\frac{\partial m'_i}{\partial t} \quad = \quad -\frac{\partial p'}{\partial x_i} - \frac{\partial m'_j u_i}{\partial x_j} - \frac{\partial m_j u'_i}{\partial x_j} + \frac{\partial \tau'_{ji}}{\partial x_j} \tag{4.79}$$

$$\frac{\partial p'}{\partial t} \quad = \quad -\frac{\partial p' u_i}{\partial x_i} - \frac{\partial p u'_i}{\partial x_i}$$

$$\begin{aligned}
&+\frac{\partial}{\partial x_i}\left(\lambda(\gamma-1)\frac{\partial T'}{\partial x_i}\right)+\frac{\partial}{\partial x_i}\left(\frac{C_p}{Pr}\frac{\partial \mu}{\partial T}T'(\gamma-1)\frac{\partial T}{\partial x_i}\right)\\
&-(\gamma-1)p'\frac{\partial u_i}{\partial x_i}-(\gamma-1)p\frac{\partial u_i'}{\partial x_i}\\
&+(\gamma-1)\tau_{ij}'\frac{\partial u_i}{\partial x_j}+(\gamma-1)\tau_{ij}\frac{\partial u_i'}{\partial x_j} \qquad (4.80)
\end{aligned}$$

with

$$m_i' = \rho' u_i + \rho' u_i \qquad (4.81)$$

$$p' = \rho' RT + \rho RT' \qquad (4.82)$$

$$\tau_{ij}' = \frac{\partial \mu}{\partial T}T' s_{ij} + \mu s_{ij}' = \frac{\partial \mu}{\partial T}T' s_{ij} + \mu\left(\frac{\partial u_i'}{\partial x_j}+\frac{\partial u_j'}{\partial x_i}-\delta_{ij}\frac{2}{3}\frac{\partial u_k'}{\partial x_k}\right), \qquad (4.83)$$

where an $'$ indicates variational quantities. Note, that because of the non-linearity of the Navier Stokes equations, equations (4.78)-(4.80) still contain non-variational quantities. Thus, those quantities have either to be saved during a precomputation of the primal flow solution or they must be computed parallel to the solution of the sensitivity equations.

4.6.3. Adjoint Equations

Following the procedure introduced in section (3.1) and neglecting boundary terms the adjoint equations corresponding to the Navier Stokes equations (4.68)-(4.70) are given by

$$\frac{\partial \rho^*}{\partial t} = u_i u_i^* + TC_V T^* \qquad (4.84)$$

$$\frac{\partial m_i^*}{\partial t} = -\frac{\partial \rho^*}{\partial x_i} - u_j\frac{\partial m_j^*}{\partial x_i} - u_i^* \qquad (4.85)$$

$$\frac{\partial p^*}{\partial t} = -\frac{\partial m_i^*}{\partial x_i} + \frac{\partial u_i^*}{\partial x_i}(\gamma-1)p^* - (\gamma-1)u_i\frac{\partial p^*}{\partial x_i} \qquad (4.86)$$

with

$$\begin{aligned}
\rho u_i^* &= m_j\frac{\partial m_i^*}{\partial x_j}+\frac{\partial \tau_{ji}^*}{\partial x_j}-(\gamma-1)\frac{\partial \tau_{ji}p^*}{\partial x_j}\\
&\quad +(\gamma-1)p\frac{\partial p^*}{\partial x_i}+\frac{\partial}{\partial x_i}\left(\frac{\partial u_i}{\partial x_i}\right)^* \qquad (4.87)
\end{aligned}$$

$$\tau_{ij}^* = \mu\left(\frac{\partial m_i^*}{\partial x_j}+\frac{\partial m_j^*}{\partial x_i}-2(\gamma-1)p^*\left(\frac{\partial u_i}{\partial x_j}+\frac{\partial u_j}{\partial x_i}\right)\right) \qquad (4.88)$$

$$\left(\frac{\partial u_i}{\partial x_i}\right)^* = -\frac{2}{3}\tau_{ii}^* + (\gamma-1)pp^* \qquad (4.89)$$

$$\rho C_V T^* \quad = \quad -s_{ij}\frac{\partial \mu}{\partial T}\frac{\partial m_i^*}{\partial x_j} + \frac{C_P}{Pr}(\gamma-1)\mu\frac{\partial^2 p^*}{\partial {x_i}^2} + s_{ij}(\gamma-1)p^*\frac{\partial \mu}{\partial T}\frac{\partial u_i}{\partial x_j}, \tag{4.90}$$

where $\cdot^*$ denotes the corresponding adjoint quantity of the primal variable.

4.6.4. LES Modelling

Scales of the turbulent motion not resolved by the numeric scheme must be modeled. Different methodologies exist in modeling such flows. One possibility is to use a statistical approach based on the Reynolds averaged Navier Stokes (RANS) equations. This, however, requires heuristic modeling of the unclosed Reynolds-stresses [105]. Common RANS models are, however, usually not adequate for aeroacoustic computations. Alternatively, only the large energy-containing scales are simulated, the small scales (subgrid scales) on the other hand are modeled. This approach is called LES. A scale separation is achieved by using a low-pass filtering procedure, such that equations for the filtered variables are solved with model terms expressed in terms of the closed variables. Well validated models based on physical reasoning exist for such LESs, such as eddy viscosity or scale similarity models [100, 62, 33]. In this work a LES model based on explicit filtering alone is chosen. The basic principle is to achieve the additional dissipation, usually resulting from LES models, with an explicit filter. In this work the 10^{th} order filter also used for numerical stabilization and described in section 4.1.2 is used. Explicit filtering has been successfully validated as a LES model for a broad range of flows such as isotropic turbulence [90], incompressible channel flow [89], supersonic channel flow [65] or subsonic jet [10, 30]. Using explicit filtering as a LES model has the advantage that the implementation of the adjoint LES model can be achieved with minimal additional effort. For the discrete adjoint the filtering of the forward equations leads to a filtering with the transposed filter operator in the adjoint equations (see equation (4.38)). For the continuous adjoint it can be rectified to filter the adjoint solution with the same filter used during the forward solution, as shown in [59]. This approach has the advantage that an elaborate analytical derivation of the adjoint LES model, as done in e.g. [21], can be avoided. A similar approach for the solution of a continuous adjoint LES was used in [50]. While a dynamic Smagorinsky model was used for the solution of the primal flow equations, the adjoint solution was stabilized using explicit filtering alone. However, using this approach further inconsistencies are introduced between the equations of motion and its adjoint.

4.7. Boundary Conditions

4.7.1. Characteristic Boundary Conditions

In order to be able to simulate flows in the absence of boundaries so called non-reflecting boundary conditions (NBC) are necessary to simulate the behavior of a quiescent medium at the computational boundaries. In this work the so called CBC [72, 57] are used for this purpose and their derivation is given for the equations (4.68)-(4.70). The approach follows the ideas explained in [98].

In areas far away from turbulent regions viscous effects are expected to play only a minor role. Thus, setting viscosity to zero can be justified outside turbulent regions and equations (4.68)-(4.70) can be written as

$$\frac{\partial \mathbf{U}}{\partial t} + \sum_{k=1}^{3} A_k \frac{\partial \mathbf{U}}{\partial x_k} + \mathbf{C} = 0, \tag{4.91}$$

where $\mathbf{U} = (\rho, m_1, m_2, m_3, p)^T$ and

$$A_1 = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ -u_1u_1 & 2u_1 & 0 & 0 & 1 \\ -u_1u_2 & u_2 & u_1 & 0 & 0 \\ -u_1u_3 & u_3 & 0 & u_1 & 0 \\ -c^2u_1 & c^2 & 0 & 0 & u_1 \end{pmatrix} \tag{4.92}$$

$$A_2 = \begin{pmatrix} 0 & 0 & 1 & 0 & 0 \\ -u_2u_1 & u_2 & u_1 & 0 & 0 \\ -u_2u_2 & 0 & 2u_2 & 0 & 1 \\ -u_2u_3 & 0 & u_3 & u_2 & 0 \\ -c^2u_2 & 0 & c^2 & 0 & u_2 \end{pmatrix} \tag{4.93}$$

$$A_3 = \begin{pmatrix} 0 & 0 & 0 & 1 & 0 \\ -u_3u_1 & u_3 & 0 & u_1 & 0 \\ -u_3u_2 & 0 & u_3 & u_2 & 0 \\ -u_3u_3 & 0 & 0 & 2u_3 & 1 \\ -c^2u_3 & 0 & 0 & c^2 & u_3 \end{pmatrix} \tag{4.94}$$

$$\mathbf{C} = 0. \tag{4.95}$$

Next, a decomposition of the matrixes A_k is introduced

$$A_k = S_k^{-1} \Lambda_k S_k, \tag{4.96}$$

where Λ_k is a diagonal matrix with the eigenvalues of A_k on its diagonal and S_k^{-1} is a matrix with the right eigenvectors of A_k as its columns (hence S_k is a matrix with the left eigenvectors of A_k as its rows). With this definition equation (4.91) can be written in characteristic form for a boundary normal to direction k

$$S_k \frac{\partial \mathbf{U}}{\partial t} = \underbrace{\Lambda_k S_k \frac{\partial \mathbf{U}}{\partial x_k}}_{a} + \underbrace{\sum_{\substack{i=1 \\ i \neq k}}^{3} S_i A_i \frac{\partial \mathbf{U}}{\partial x_i}}_{b} + S\mathbf{C} = 0. \tag{4.97}$$

It can be shown that term a in equation (4.97) can be interpreted as a traveling wave with wave speeds corresponding to the eigenvalues of A_k. Thus, we can conclude that non-reflecting boundary conditions can be achieved by solving equation (4.97) with incoming waves set to zero. This can be achieved by setting those eigenvalues of Λ_k to zero which have a different sign than the boundary normal.

It should be noted, that the above procedure is exact for a hyperbolic system in one dimension, only. The treatment of term b in equation (4.97) is not exact in terms of characteristics. Although it is not clear how to handle this term, recent research suggests that it is better to leave it unchanged [57].

Instead of setting the ingoing characteristics to zero to obtain non-reflecting boundary conditions the characteristics can also be set explicitly to force some prescribed solution at this boundary. This is done in this work at the inflow boundary of the jet.

4.7.1.1. Characteristics of Navier Stokes Flow

In explicit form the characteristic equations (4.97) can be written as

$$\frac{\partial m_j}{\partial t} - \frac{u_j}{c^2} + L_{ij} + \ldots = 0 \quad \forall j \in \{1,2,3\}\backslash i \tag{4.98}$$

$$u_i \frac{\partial \rho}{\partial t} - \frac{u_i}{c^2} + L_{ii} + \ldots = 0 \tag{4.99}$$

$$\frac{1}{2}\frac{\partial p}{\partial t} - \frac{\rho c}{2}\frac{\partial u_i}{\partial t} + L_{i4} + \ldots = 0 \tag{4.100}$$

$$\frac{1}{2}\frac{\partial p}{\partial t} + \frac{\rho c}{2}\frac{\partial u_i}{\partial t} + L_{i5} + \ldots = 0, \tag{4.101}$$

where $\ldots$ denote part b in equation (4.97) and will not be given explicitly. The characteristic wave amplitudes $\mathbf{L}_i \equiv \Lambda_k S_k \frac{\partial \mathbf{U}}{\partial x_k}$ are given by

$$L_{ij} = \lambda_j \left(\frac{\partial m_j}{\partial x_i} - \frac{u_j}{c^2}\frac{\partial p}{\partial x_i} \right) + F_j \quad \forall j \in \{1,2,3\}\backslash i \tag{4.102}$$

$$L_{ii} = \lambda_i u_i \left(\frac{\partial \rho}{\partial x_i} - \frac{1}{c^2}\frac{\partial p}{\partial x_i} \right) + F_i \tag{4.103}$$

$$L_{i4} = \frac{\lambda_4}{2} \left(\frac{\partial p}{\partial x_i} - c\rho \frac{\partial u_i}{\partial x_i} \right) + F_4 \tag{4.104}$$

$$L_{i5} = \frac{\lambda_5}{2} \left(\frac{\partial p}{\partial x_i} + c\rho \frac{\partial u_i}{\partial x_i} \right) + F_5, \tag{4.105}$$

where $\lambda = (u_i, u_i, u_i, u_i - c, u_i + c)$ are the eigenvalues of S_i. The forcings F_k are chosen to mimic the temporal derivatives of the characteristic equations (4.98)-(4.101)

$$F_j = f_j \left(m_j - m_{j,f} - \frac{u_j}{\gamma} \left(\rho - \rho_f \frac{T_f}{T} \right) \right) \quad \forall j \in \{1,2,3\}\backslash i \tag{4.106}$$

$$F_i = f_i u_i \left(\rho - \rho_f - \frac{1}{\gamma}\left(\rho - \rho_f \frac{T_f}{T}\right)\right) \tag{4.107}$$

$$F_4 = \frac{f_4}{2}\left(p - p_f - \rho c\left(u_i - u_{i,f}\right)\right) \tag{4.108}$$

$$F_5 = \frac{f_5}{2}\left(p - p_f + \rho c\left(u_i - u_{i,f}\right)\right), \tag{4.109}$$

where f_k are parameters determining the strength of the inflow forcing. The forcing should be applied for the case of ingoing waves only ($f_k = 0$ for outgoing waves). The choices of F_k are to some degree arbitrary and other choices are possible.

The PDE used for discretization in this work is obtained by multiplying equation (4.98)-(4.101) by S_i^{-1} and read

$$\frac{\partial \rho}{\partial t} = -\frac{L_{ii}}{u_i} - \frac{1}{c^2}\left(L_{i4} + L_{i5}\right) - \sum_{\substack{k=1\\ i\neq k}}^{3}\left(\frac{\partial m_k}{\partial x_k}\right) \tag{4.110}$$

$$\begin{aligned}\frac{\partial m_j}{\partial t} =& - L_{ij} - \frac{u_j}{c^2}\left(L_{i4} + L_{i5}\right) - \frac{\delta_{ij}}{c}\left(-L_{i4} + L_{i5}\right)\\ & - (1-\delta_{ij})\frac{\partial p}{\partial x_j} - \sum_{\substack{k=1\\ i\neq k}}^{3}\frac{\partial u_j m_k}{\partial x_k}\end{aligned} \tag{4.111}$$

$$\frac{\partial p}{\partial t} = - L_{i4} - L_{i5} - \sum_{\substack{k=1\\ i\neq k}}^{3}\left(u_k \frac{\partial p}{x_k} + \gamma p \frac{\partial u_k}{x_k}\right). \tag{4.112}$$

4.7.1.2. Characteristics of Adjoint Equations

The discrete adjoint is uniquely defined by the choice of the discretization of the Navier-Stokes equations. However, the same is not true for the continuous adjoint. Thus, the boundary conditions for the continuous adjoint used in this work are explained next. In principle one could calculate the exact adjoint equations of the inner scheme (4.68)-(4.70) together with the boundary conditions (4.110)-(4.112). However, as the obtained equations become very cumbersome and unhandy another approach was chosen. For this work first the adjoint system of equations (4.91) is derived and then the characteristics of this system are obtained. This approach is similar to the adjoint boundary conditions used in [103, 50].

This approach can be justified by physical means in the sense that reflections of the adjoint solution should be reduced at the boundaries. However, it should be noted that this approach does not represent the exact adjoint of equations (4.68)-(4.70) and (4.110)-(4.112) and inconsistencies between the Navier Stokes equations and their adjoint parts are introduced. For example the boundary terms arising from the integration by parts of the inner terms, done during the derivation of the continuous adjoint system, are not taken into account. Furthermore, the transposed finite difference operators present in the discrete adjoint are effectively substituted with their non transposed operators (or their negative, depending on the physical meaning of the linear operator) in the continuous adjoint approach. This substitution is rectified in the inner region, where the FD operators are almost symmetric due to the central differencing and a low grid stretching. In this work strong grid stretching and non central FD are used at the boundary, thus the errors connected with the continuous adjoint approach are expected to be larger near the boundaries than in the inner region of the computational domain. This is discussed further in section 6.4.

Neglecting boundary terms the adjoint of equations (4.91) is given by

$$\frac{\partial \xi}{\partial t} + \sum_{k=1}^{3} \left[A_k^T \frac{\partial \xi}{\partial x_k} + \underbrace{\frac{\partial A_k^T}{\partial x_k}\xi}_{a} - \underbrace{\left(\nabla_{\mathbf{U}} \xi^T A_k\right) \frac{\partial \mathbf{U}}{\partial x_k}}_{b} \right] - \underbrace{\left(\nabla_{\mathbf{U}} \mathbf{C}^T\right) \xi}_{c} \tag{4.113}$$

where $\nabla_{\mathbf{U}} = \left(\frac{\partial}{\partial \rho}, \frac{\partial}{\partial m_1}, \frac{\partial}{\partial m_2}, \frac{\partial}{\partial m_3}, \frac{\partial}{\partial p}\right)^T$. From equation (4.113) it becomes obvious that the characteristics of the adjoint equation are given by $\widetilde{L} \equiv \Lambda_k \widetilde{S}_k \frac{\partial \xi}{\partial x_k}$, where $\widetilde{S}_k$ is a matrix with the right eigenvectors of A_k^T as its rows. Note, that the eigenvalues of A^T and A are equal for any matrix. However, as the adjoint equations have to be solved backward in time the sign changes for the effective eigenvalues during the time integration (inflow boundaries become outflow boundaries and vice versa).

In a more explicit form the adjoint characteristic wave amplitudes read

$$L_{ij}^* = \lambda_j \frac{\partial m_j^*}{\partial x_i} \tag{4.114}$$

$$L_{ii}^* = -\frac{\lambda_i}{c^2}\left(\frac{\partial \rho^*}{\partial x_i} + \sum_{j=1}^{3} u_j \frac{\partial m_j^*}{\partial x_i}\right) \tag{4.115}$$

$$L_{i4}^* = \frac{\lambda_4}{2}\left(\frac{\partial p^*}{\partial x_i} + \frac{1}{c^2}\frac{\partial \rho^*}{\partial x_i} + \frac{u_i - c}{c^2}\frac{\partial m_i^*}{\partial x_i} + \sum_{\substack{t=1\\ t\neq i}} \frac{u_t}{c^2}\frac{\partial m_t^*}{\partial x_i}\right) \tag{4.116}$$

$$L_{i5}^* = \frac{\lambda_5}{2}\left(\frac{\partial p^*}{\partial x_i} + \frac{1}{c^2}\frac{\partial \rho^*}{\partial x_i} + \frac{u_i + c}{c^2}\frac{\partial m_i^*}{\partial x_i} + \sum_{\substack{t=1\\ t\neq i}} \frac{u_t}{c^2}\frac{\partial m_t^*}{\partial x_i}\right). \tag{4.117}$$

One could calculate equation (4.113) at the boundaries with ingoing waves set to zero, in order to obtain non reflecting boundary conditions. However, for the sake of implementational simplicity in this work equations (4.84)-(4.86) are calculated at the boundaries and subsequently the ingoing characteristics of (4.114)-(4.117) are subtracted. This way the terms *a*-*c* in equation (4.113) don't have to be calculated explicitly.

Again, it should be noted that this boundary treatment is based on the physical idea of reducing reflections at the boundary and is not consistent with the boundary treatment of the Navier Stokes equations. For lack of alternatives and due to successful applications of similar approaches in [103, 84] a similar approach was adopted in this work.

4.7.2. Sponge Regions

Although the boundary conditions introduced in the former sections have proven to be effective, some approximations were necessary for their derivation. Thus, reflections are still present at the boundaries and additional sponge regions have to be applied to reduce the reflections further.

The idea of a sponge region is to force a solution u to some prescribed solution u_f with

$$\frac{\partial u}{\partial t} = \ldots - f(\mathbf{x})(u - u_f), \tag{4.118}$$

where $\ldots$ denotes the RHS of the PDE and $f(\mathbf{x})$ is a function which is zero where no sponging is necessary and positive in the sponge regions. The value of u_f may have a spatial or temporal dependency and its choice must be based on physical intuition.

Analytical solutions in one dimension suggest that reflections are damped stronger with increasing forcing value in the sponge regions [8]. However, this is not true in three dimensions. Furthermore, the sponge forcing becomes stiff for high forcing amplitudes, making the computation inefficient. [58] suggests a strategy to estimate the necessary sponge parameters depending on the wavelength and orientation of perturbations entering the sponge zones. However, as these perturbations can have a broad spectrum and are usually not known the determination of the sponge parameters has to be based on experience and trial and error. The precise shape of the sponge function $f(\mathbf{x})$ will be presented in section 5.3.

4.7.3. Further Boundary Treatment

Despite the methods introduced in the previous sections some other techniques have been used at the boundaries to further reduce reflections and retain numerical stability. These techniques are explained now.

Filtering In the sponge regions a sixth order filter is applied additionally to the high order filter used for numerical stability. This low order filter is intended to dissipate small scale structures and thus reduces the degree of turbulence at the boundaries.

Grid Stretching At some boundaries the grid is stretched. Here, the grid becomes coarse so that small scales can not be resolved anymore and are dissipated, hopefully. A strong grid stretching near the boundaries also implies that less grid points are needed in the sponge regions, where the flow solution is unphysical anyway. The specific grid stretchings used in this work are given in section 5.2.

temporally moving average An average value $\bar{\mathbf{m}}$ is updated according to $\bar{\mathbf{m}} = \alpha\bar{\mathbf{m}} + (1-\alpha)\bar{\mathbf{u}}$ with some α smaller but near to one, where $\bar{\mathbf{u}}$ are the flow variables averaged in periodic directions. This moving average is used as a target value for the sponge forcing at the outflow, for flow variables where the choice of target values is not clear, namely the velocities in stream- and shearwise directions. Similar to the sponge parameters appropriate values of α have to be determined by trial and error.

5. Physical and Numerical Setup

Throughout this work the system of choice is a plane jet at a Mach number of $Ma = 0.9$ and a Reynolds number of $Re = 2000$, based on jet inflow velocity and width. Typical reference values together with its non dimensional values are shown in table 5.1 for a jet with air at room temperature. Experiments at high Mach-numbers are usually performed at much higher Reynolds numbers of $Re = 10^5$ or higher [79, 95]. The reason for this is given in table 5.1. With the given reference values a jet in air would require a slot width of $D \approx 0.1mm$. Such a miniature jet would be experimentally unhandy. Nevertheless, a rather low Reynolds number was chosen for this work as it allows simulations where the physical dissipation exceeds the numerical dissipation. Such well resolved cases are of interest as it was shown in [22] that for the control of the turbulent kinetic energy in a channel flow the optimization is more effective using a DNS resolution, compared to LES. Furthermore, results on optimal flow control in [24] suggest that the controllability decreases with increasing Reynolds number.

5.1. Optimization Setup

5.1.1. Cost Functional

For the purpose of sound reduction a measure for the noise emission of a jet is required. In this work the cost functional we aim to minimize is defined as a weighted integral over the pressure fluctuations

$$\Im = \int_T \int_{\hat{\Omega}} r(\mathbf{x}, t) \left(p(\mathbf{x}, t) - \bar{p}(\mathbf{x})\right)^2 dV dt, \qquad (5.1)$$

ref-value	meaning	value	non-dimensio-nalization	dimension-less value
ρ_{ref}	mass density	$1.16\frac{kg}{m^3}$	$\frac{1}{\rho_{ref}}$	1
T_{ref}	temperature	$295\ K$	$\frac{1}{T_{ref}}$	1
$C_{p,ref}$	heat-capacity at constant pressure	$1000\frac{J}{kgK}$	$\frac{T_{ref}}{u_{ref}^2}$	≈ 3.0864
μ_{ref}	dynamic viscosity	$18.48 \cdot 10^{-6} Pa\ s$	$\frac{1}{\rho_{ref} u_{ref} l_{ref}}$	$5 \cdot 10^{-4}$
γ	ratio of specific heats	$\frac{7}{5}$	1	$\frac{7}{5}$
Pr	Prandtl-number	0.71	1	0.71
Ma	Mach-number	0.9	1	0.9
Re	Reynolds-number	2000	1	2000
T_{suth}	Sutherland temperature	$110.4\ K$	$\frac{1}{T_{ref}}$	$\frac{110.4}{295}$
$R_{ref} = C_{p,ref}\frac{\gamma-1}{\gamma}$	gas constant	$\approx 285.7\frac{J}{kgK}$	$\frac{T_{ref}}{u_{ref}^2}$	≈ 0.8818
$C_{v,ref} = \frac{C_{p,ref}}{\gamma}$	heat-capacity at constant volume	$714.3\frac{J}{kgK}$	$\frac{T_{ref}}{u_{ref}^2}$	≈ 2.205
$c_{ref} = \sqrt{\gamma R_{ref} T_{ref}}$	speed of sound	$\approx 343.5\frac{m}{s}$	$\frac{1}{u_{ref}}$	$\frac{1}{Ma}$
$u_{ref} = Ma\ c_{ref}$	velocity	$\approx 309.2\frac{m}{s}$	$\frac{1}{u_{ref}}$	1
$p_{ref} = \rho_{ref} R_{ref} T_{ref}$	pressure	$\approx 98.11 kPa$	$\frac{1}{\rho_{ref} u_{ref}^2}$	≈ 0.8818
$l_{ref} = \frac{Re\ \mu_{ref}}{\rho_{ref} u_{ref}}$	length	$\approx 1.027 \cdot 10^{-4} m$	$\frac{1}{l_{ref}}$	1
$t_{ref} = \frac{l_{ref}}{u_{ref}}$	time	$\approx 3.321 \cdot 10^{-7} s$	$\frac{u_{ref}}{l_{ref}}$	1

Table 5.1: List of all reference values used throughout this work, together with non-dimensionalization and non-dimensional reference values. The reference values above the horizontal line can be chosen independently, while the values below this line are determined by previous reference values.

where T is the optimization interval, $\hat{\Omega}$ is the complete computational domain and

$$\bar{p}(\mathbf{x}) = \frac{1}{\int_T \int_0^{L_y} r(\mathbf{x},t)dtdy} \int_T \int_0^{L_y} r(\mathbf{x},t)p(\mathbf{x},t)dtdy, \tag{5.2}$$

is an average over time and the periodic spanwise direction. $r(\mathbf{x})$ is a weighting function that is one in an area Ω in the farfield and zero everywhere else. To avoid aliasing a

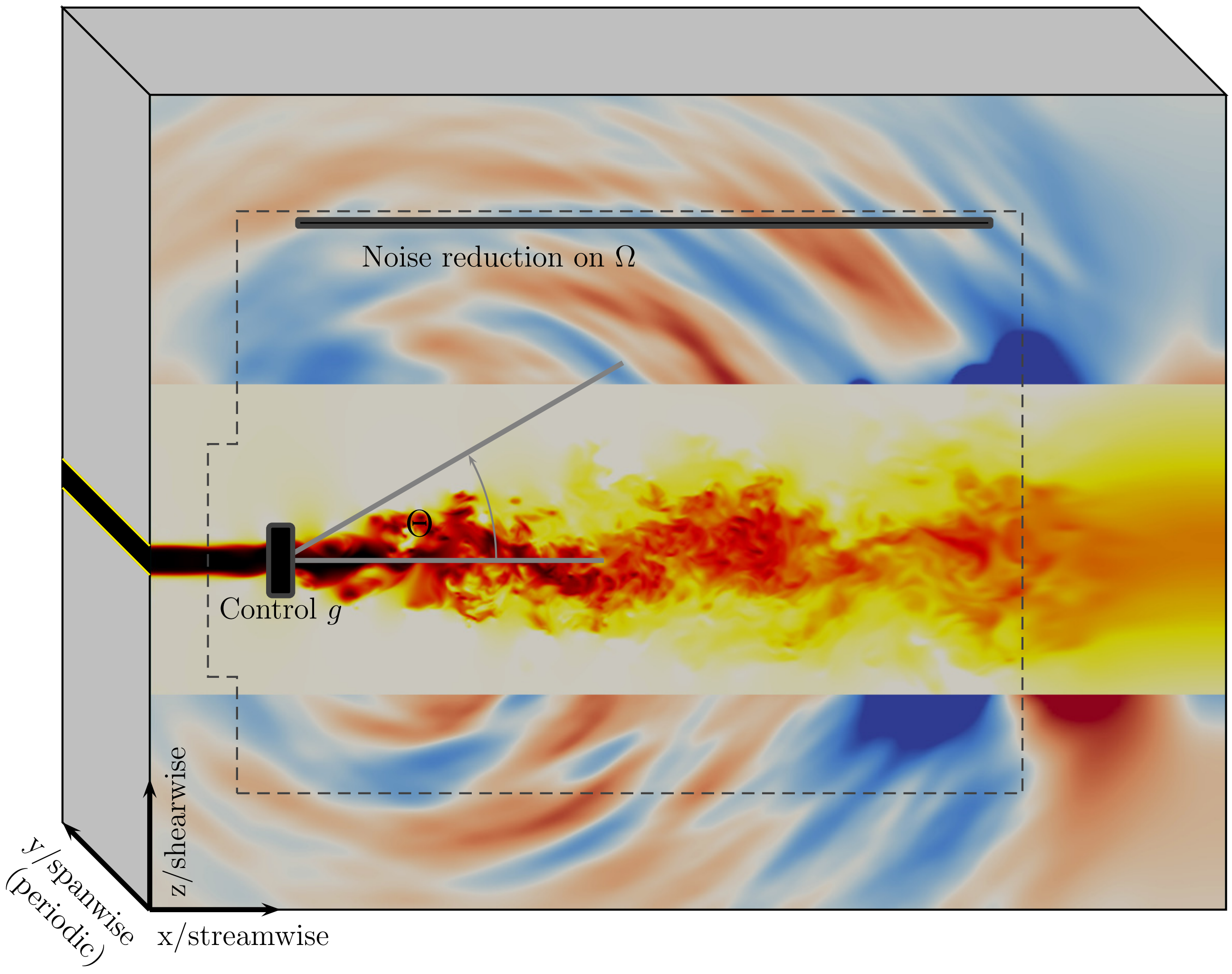


Figure 5.1: Illustration of the control setup. The jet is forced in a small volume within the shearlayers of the jet. The noise reduction is expected to take place in the observer region in the near farfield of the jet, indicated by a thick black horizontal line. Shown is the streamwise velocity component in the turbulent region and the density in the farfield. The areas outside the dashed lines are unphysical sponge regions.

hyperbolic tangent profile is applied at the boundaries of Ω. The jet geometry together with the position of the noise reduction zone Ω is illustrated in figure 5.1.

5.1.2. Control

The jets are controlled using a volume forcing, which is achieved by adding forcing terms to the equations of motion

$$\frac{\partial \rho}{\partial t} = \ldots + \qquad s_\rho(\mathbf{x}, t) g_\rho(\mathbf{x}, t) \tag{5.3}$$

$$\frac{\partial m_i}{\partial t} = \ldots + \qquad s_{m_i}(\mathbf{x},t) g_{m_i}(\mathbf{x},t) \tag{5.4}$$

$$\frac{\partial p}{\partial t} = \ldots + \qquad \rho R s_T(\mathbf{x},t) g_T(\mathbf{x},t), \tag{5.5}$$

where ... denote the RHS of equations (4.68)-(4.70), $g.$ are the space and time dependent controls and $s.$ are window functions to ensure a smooth transition from uncontrolled to controlled areas. The window functions are given by

$$s(\mathbf{x},t) = s_{\text{window}}(x, 2\Delta_x) s_{\text{window}}(z, 2\Delta_z) s_{\text{window}}(t, 5\Delta_t) \tag{5.6}$$

$$s_{\text{window}}(k,\Delta) = \frac{1}{2}\left(\operatorname{erf}\left(\left(k - k_{\text{start}} - 2\Delta\right)/\Delta\right) - \operatorname{erf}\left(\left(k - k_{\text{end}} + 2\Delta\right)/\Delta\right)\right), \tag{5.7}$$

where $\operatorname{erf}(x)$ is the error function, $\Delta.$ is the grid spacing for spatial and the timestep for temporal directions and k_{start} and k_{end} are the start and end positions/times of the controlled area. With this definitions the controls can be interpreted as density, momentum and temperature forcing, respectively. The control is positioned in an area from four to five jet diameters away from the inflow boundary. In the shearwise direction the control area has a length of $2.5D$ and is centered around the jet centerline. The control spans over the whole spanwise direction. The position of the control is illustrated in figure 5.1. The control is given every n^{th} RK iteration. The control values in-between are obtained using linear interpolation. This interpolation determines the weights $\gamma_{s,i}$ in equation (4.32).

It should be noted that this kind of control can in this form not be applied to experiments. The same holds for the adjoint optimization procedure, as the adjoint computation is too expensive to be performed in real time. Furthermore, in experiments one can not assume knowledge of the phase space over the whole optimization interval. This work concentrates on the applicability and effectivity of the adjoint optimization for a non linear time dependent and three dimensional flow setup. Furthermore, results from optimal control computations might serve as a benchmark for easier applicable but suboptimal control strategies.

5.1.3. Gradient Computation

Having defined the control the gradient is uniquely defined by equation (4.42) for the discrete adjoint. With a control given at discrete times and interpolation in-between the summation in equation (4.42) has to be substituted with a temporal integral for the continuous adjoint

$$\left(\frac{d\Im}{dg_i}\right)^T = \int dt\ \gamma_i(t)\xi(t), \tag{5.8}$$

where $\xi(t)$ is the continuous adjoint and the weights $\gamma_i(t)$ are now continuous functions of time. As there isn't a unique way of discretizing this integral ambiguities arise for the gradient computation. In this work the gradient is computed by assuming that the adjoint

Case	L_x	L_y	L_z	n_x	n_y	n_z	$\Delta_{x,min}$	$\Delta_{y,min}$	$\Delta_{z,min}$	Δt
DNS2D	30	-	34	512	1	640	0.04	-	0.036	0.017
ELES3D	37	9	28	416	64	320	0.071	0.14	0.065	0.03
LES3D	37	9	28	512	160	400	0.051	0.056	0.051	0.021
DNS3D	37	9	28	800	288	600	0.029	0.031	0.028	0.012

Table 5.2: Parameters of the plane jet simulations considered in this work. The domain lengths L_i were normalized by the jet diameter D. The Reynolds number is $Re = U_j \rho_j D/\mu_j = 2000$ and the Mach number is $Ma = U_j/c_j = 0.9$ for all simulations. The number of grid points in the respective coordinate directions are represented by n_i. $\Delta_{i,min}$ gives the minimum grid-spacing in direction i. Δt gives the time step used during the optimization computations nondimensionalized by D/U_j. The subscript j denotes mean values at the jet inflow.

variables are nearly constant over short time intervals, leading to

$$\left(\frac{d\Im}{dg_i}\right)^T = \int dt\ \gamma_i(t)\xi(t) \approx \xi(t_i) \int dt\ \gamma_i(t), \tag{5.9}$$

where t_i is defined by $\gamma_i(t_i) = 1$. In this work a control is given every second or third RK iteration for the continuous adjoint cases. It will be shown that correlations of the flow fields and thus the adjoint solutions are high over such time intervals. Furthermore, while the errors introduced by the discretization of the continuous adjoint accumulate during the RK iteration the error introduced with equation (5.9) keeps constant with time. Because of this we expect the error introduced with equation (5.9) to be negligible compared to the errors introduced by the discretization of the continuous adjoint.

5.2. Computational Grids

To be able to study the dependence of the grid resolution on the optimization procedure the jet simulation is carried out on different grids. The simulations consist of a two dimensional DNS, two three dimensional LESs with a finer and a coarser resolution and a three dimensional plane jet with a DNS-like resolution. These cases will be referred to as DNS2D, LES3D, ELES3D and DNS3D, respectively. The cases are listed in table 5.2, together with some of its numerical parameters. It is well known that the two dimensional Navier Stokes equations reveal different physics compared to its three dimensional counterpart [88]. One important physical mechanism missing is, for example, the vortex stretching. Nevertheless, case DNS2D is useful for several tests and numerical validation due to its low computational requirements as opposed to three dimensional computations and the various simulations performed in the literature, which make comparisons possible.

An orthogonal cartesian grid with grid stretching in the non-periodic directions is used. The grid is exemplarily illustrated in figure 5.3. Note that the grid is not shown in spanwise direction as no grid stretching was applied in this periodic direction.

Figure 5.3 shows the grid spacing $\Delta x_i = x_{i+1} - x_i$ and relative grid stretching $\Delta^2 x_i =$

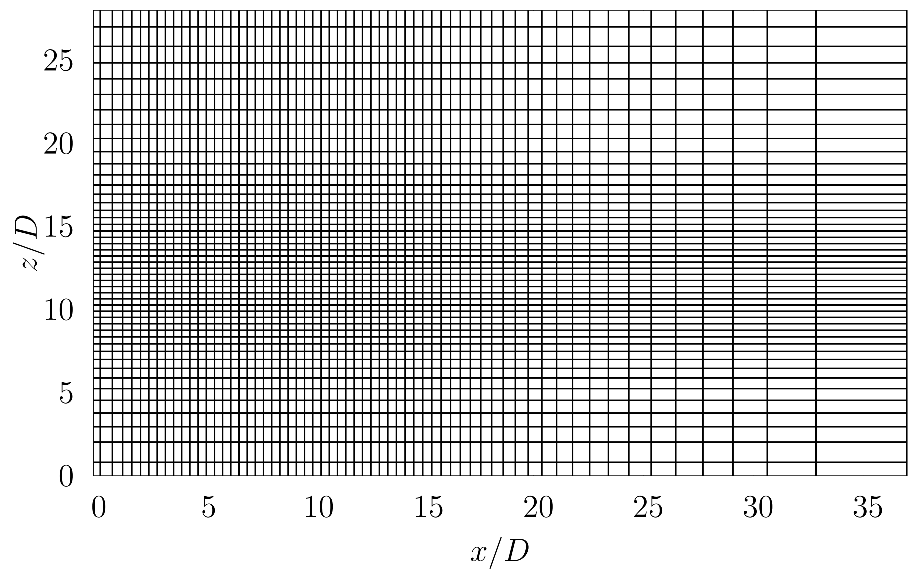


Figure 5.2: Illustration of the grid used for the plane jet simulations. Shown is every 13th grid point in stream- and shearwise direction for case DNS3D.

$\frac{2(x_{i+1}-2x_i+x_{i-1})}{x_{i+1}-x_{i-1}}$ in streamwise direction, where x_i is the i-th grid point in streamwise direction. The grid stretching is chosen such that the resolution is highest for $x < 15$, which includes the area of transition to turbulence and the core breakdown, which is the dominant region responsible for sound production. Further downstream the grid spacing is increased with a grid stretching less than 0.5%. About seven jet diameters before the end of the computational domain a strong grid stretching is applied to reduce the number of grid points in the unphysical outflow sponge region.

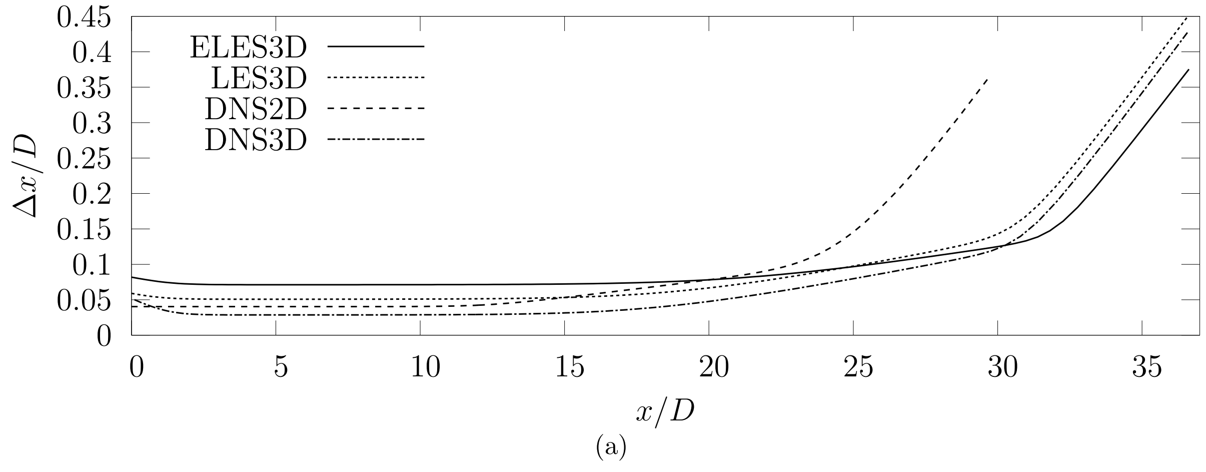
The grid has a constant grid spacing in the periodic spanwise direction with an extent of $L_y = 9$. Figure 5.4 shows the correlation of density, velocities and temperature in spanwise direction. It can be observed that the correlations decay to zero for the chosen spanwise extent and thus the jet can be expected to develop fully three dimensional turbulence.

The grid spacing and stretching in shearwise direction is shown in figure 5.5. The resolution is highest around the jet centerline with a moderate grid stretching in the near farfield of the jet. Again, a strong stretching is applied in the sponge regions near the boundaries.

Figure 5.6(a) shows an estimate of the Kolmogorov length scales η obtained from cases ELES3D, LES3D and DNS3D along the jet center line. The estimate was calculated via [73]

$$\eta = \left(\frac{\nu^3}{\epsilon}\right)^{1/4} \tag{5.10}$$

$$\epsilon = \nu\overline{\frac{\partial\widehat{u}_i}{\partial x_j}\frac{\partial\widehat{u}_i}{\partial x_j} + \frac{\partial\widehat{u}_i}{\partial x_j}\frac{\partial\widehat{u}_j}{\partial x_i}} \tag{5.11}$$



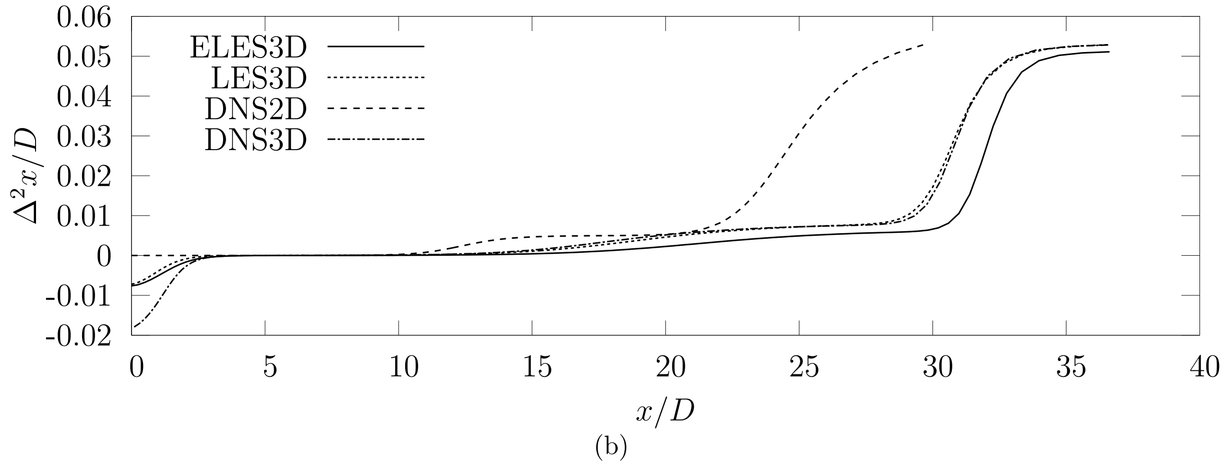


Figure 5.3: (a) Grid spacing and (b) relative grid stretching in streamwise direction for the cases listed in tabular 5.2.

$$\nu = \frac{\mu(T_{ref})}{\rho_{ref}}, \tag{5.12}$$

where $\overline{\cdots}$ denote the Reynolds average and $\widehat{u} = u - \overline{u}$. It should be noted that equation (5.10) is derived from the incompressible Navier-Stokes equations. However, as unheated jets are considered in this work, only, the density fluctuations are expected to be small enough for equation (5.10) to give a usefull approximation, anyway.

To no surprise the coarsest grid (case ELES3D) gives the largest estimate for the Kolmogorov length. The estimates of cases LES3D and DNS3D, however, agree quite well, indicating that these lengths scale might be grid converged. The minimal length observed in figure 5.6(a) is about $\eta = 0.013D$. [11] reported a Kolmogorov length of $\eta = 0.0116$ for $Re = 1700$ and $\eta = 0.0091$ for $Re = 2500$ for a round jet. Having in mind the different geometries and the uncertainties involved in determining the Kolmogorov length scale the

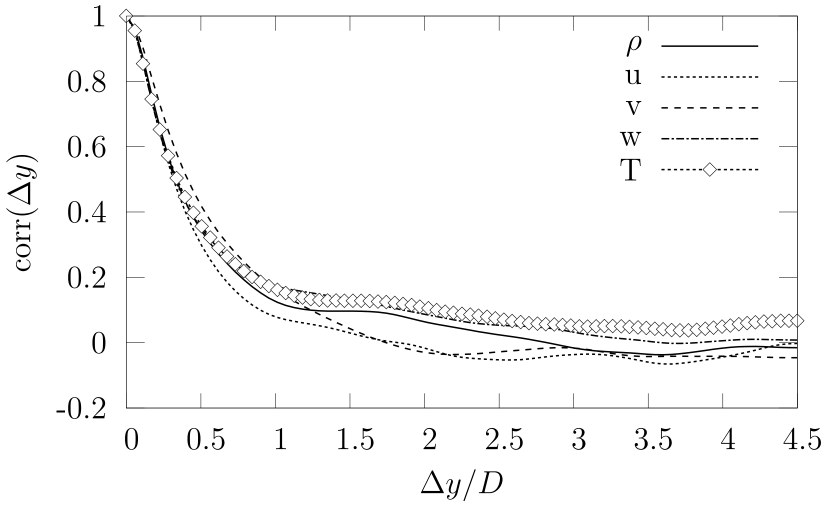


Figure 5.4: Spanwise correlation of density, velocities and temperature from case LES3D at $x = 12$ and $z = 12$. The spanwise extent L_y of the computational domain is sufficiently long for the correlations to become zero.

agreement is reasonable. The ratio between grid spacing and the Kolmogorov lengths is shown in figure 5.6(b) for the three different resolutions. Due to the larger estimation of case ELES3D the cases ELES3D and LES3D reveal similar ratios (about 4.5 for ELES3D and about 4.0 for LES3D). However, the ratio reduces to ≈ 2.5 for case DNS3D. This ratio is small enough to rectify to call case DNS3D a DNS. It can be observed that the grid spacing Kolmogorov length ratio reveals similar values in the region just before the outflow sponge region at $x \approx 30$. This is in accordance with the observation that all three cases have a similar grid spacing in this area (see figure 5.3). Figure 5.6(b) indicates that case DNS3D isn't a DNS for $x \gtrsim 25$. However, as the dominant regions of noise generation in a jet (shearlayer development, core break down) appear before this point, case DNS3D might still be regarded as a DNS.

The timestep was determined using the convective criteria

$$\Delta t = \min_{i,d\in\{x,y,z\}} \left(\mathrm{CFL} \frac{\Delta d_i}{(c_i + |u_{d,i}|)} \right), \tag{5.13}$$

where Δd_i is the local grid spacing, $u_{d,i}$ is the velocity at the i^{th} grid point in direction d and c_i is the local speed of sound at grid point i. To achieve time accuracy and to reduce the dispersion and dissipation errors associated with the temporal discretization the Courant-Friedrichs-Lewy (CFL) number was set to a moderate value of CFL $= 1$. The timesteps used during the optimization computations are listed in table 5.2.

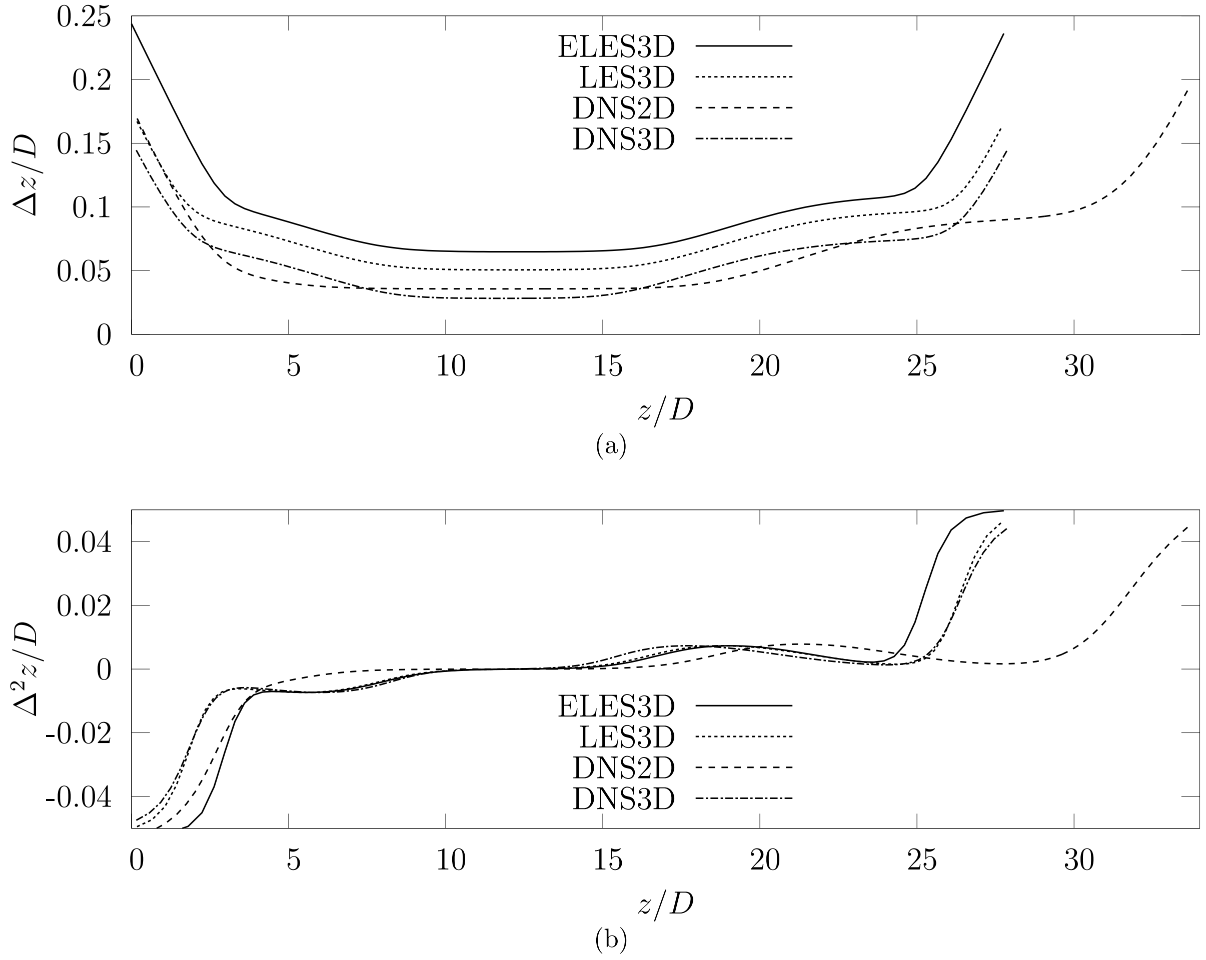


Figure 5.5: (a) Grid spacing and (b) relative grid stretching in shearwise direction for the cases listed in tabular 5.2.

5.3. Boundary Treatment

In section 4.7 the general methodology for the treatment of the boundaries was introduced. A detailed description of the specific choices made for the boundary treatment in this work will be presented in this section. For all boundaries the transition from unsponged to sponged regions was achieved by a hyperbolic tangent profile

$$f(x) = \frac{s_{amp}}{2}\left(1 \pm \tanh\left(\frac{x - s_{pos}}{s_{width}}\right)\right), \tag{5.14}$$

where $f(x)$ is the local sponge strength according to section 4.7.2. s_{amp}, s_{pos} and s_{width} are user defined parameters and the sign in front of the tanh determines whether regions

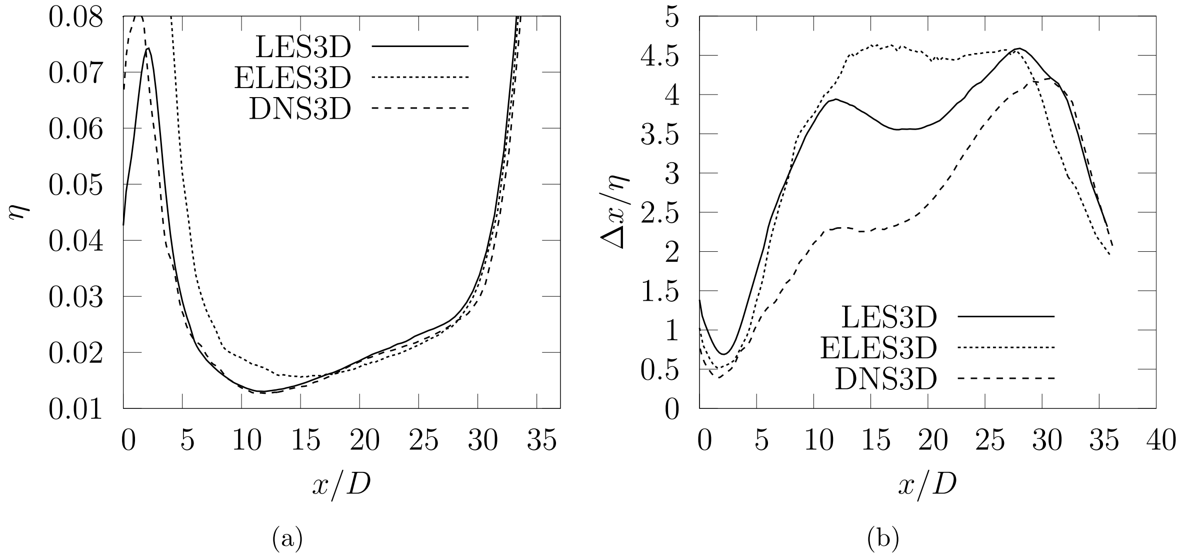


Figure 5.6: (a) Kolmogorov length scale estimated using equation (5.10) and data from cases ELES3D, LES3D and DNS3D. (b) Ratio of grid spacing and Kolmogorov length obtained from the differently resolved cases.

boundary	parameter	DNS2D	ELES3D	LES3D	DNS3D
inflow near unsponged	s_{amp}	0.06	-	0.05	0.02
	s_{pos}	1.00	-	1.25	1.00
	s_{width}	1.00	-	1.25	1.00
inflow near sponged	s_{amp}	-	0.60	0.60	0.55
	s_{pos}	-	2.00	2.00	2.00
	s_{width}	-	2.00	1.00	1.00
inflow far	s_{amp}	0.10	0.40	0.10	0.30
	s_{pos}	2.00	0.50	1.00	3.00
	s_{width}	1.00	0.50	1.00	2.00
outflow	s_{amp}	0.40	0.50	0.50	0.50
	s_{pos}	3.50	4.00	4.00	4.00
	s_{width}	3.50	3.50	3.50	3.00
lateral	s_{amp}	0.40	0.50	0.50	0.40
	s_{pos}	2.00	2.00	2.00	2.00
	s_{width}	2.00	2.00	2.00	2.00

Table 5.3: Sponge parameters for the different cases and different boundaries. See equation (5.14) for the definition of the sponge parameters.

with $x < s_{pos}$ or $x > s_{pos}$ include sponges. The specific sponge parameters for the different cases and boundaries are listed in table 5.3.

5.3.1. Inflow Boundary

The laminar profile enforced at the inflow is given by

$$\begin{aligned} U =& \frac{1}{2}(U_j - U_{co}) \left(\tanh\left(-\frac{z - (z_{center} + D/2)}{h} \right) \right. \\ & \left. + \tanh\left(\frac{z - (z_{center} - D/2)}{h} \right) \right) + U_{co}, \end{aligned} \tag{5.15}$$

where U_j is the jet velocity and D the jet diameter at the inflow. This inflow profile has become a common choice for jet simulations [30, 31]. The shear thickness was chosen to be $h = 0.10D$ for cases DNS2D, LES3D and DNS3D. Due to the coarse resolution of case ELES3D, h was set to 0.12 for this case. It is well known that the initial shear thickness has a considerable influence on the development of the jet. Thus, one should keep in mind this difference when comparing case ELES3D with cases LES3D and DNS3D. The co-flow velocity is $U_{co} = 0.01U_j$ for the three dimensional cases and $U_{co} = 0.05$ for case DNS2D. The different co-flow velocity complicates the comparison between the three- and two dimensional cases. However, as mentioned earlier, three- and two dimensional cases reveal different physics anyway and the two dimensional case primarily serves as a test and validation case. Pressure and density were forced towards their values at infinity and span- and shearwise velocities towards zero.

A precursor simulation with periodic streamwise direction was performed. The initial condition was the laminar inflow profile according to equation 5.15 but with a lower shear thickness together with a random perturbation with a prescribed powerspectrum as in [87]. This initial flow field was advanced forward in time to achieve a developement of the shearlayers towards turbulence. Figure 5.7 shows two slices of such a precursor field exemplarily for case DNS3D. For each grid a seperate precursor simulation was performed. Within a distance of two diameters away from the jet centerline the fluctuating parts of this solution were added to the laminar inflow profile in the jet simulations to speed up the transition to turbulence.

The reference velocity for the sponge at the inflow is set to the laminar inflow profile. Different sponge strengths were applied for areas near the jet centerline and areas with some distance from the jet. The flow solution was forced only weakly away from the jet centerline. In the vicinity of the jet inflow either a weak or a strong sponging is applied. In the case of a strong inflow sponging perturbations were added in this sponge zone so that further downstream the jet statistics stay about the same for both cases.

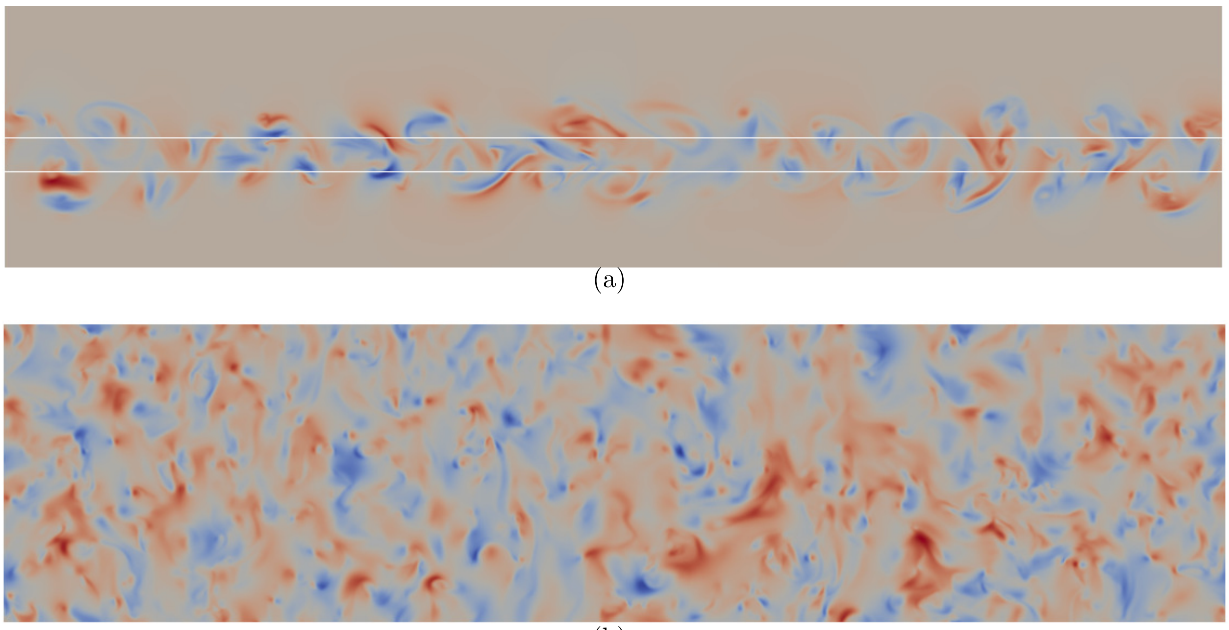


Figure 5.7: Spanwise velocity at slices in (a) streamwise and shearwise and (b) streamwise and spanwise directions of the precursor field for case DNS3D used to trigger turbulence in the jet simulations. The white horizontal lines mark the position of the shearlayers. The plane in (b) is positioned in the upper shearlayer.

5.3.2. Lateral Boundary

As non-reflective boundaries are desired at the lateral boundaries the forcing through the CBC is set to zero. Because of the entrainment the lateral boundaries are treated as inflow boundaries. Pressure, density, streamwise velocity and spanwise velocity are forced towards p_∞, ρ_∞, U_{co} and zero in the sponge, respectively. The shearwise velocity, perpendicular to the lateral boundary, is not forced by the sponge to allow entrainment.

5.3.3. Outflow Boundary

Near the outflow pressure and density are forced to their values at infinity and the spanwise velocity to zero. The remaining two velocities are sponged towards values obtained from a moving average as described in section 4.7.3.

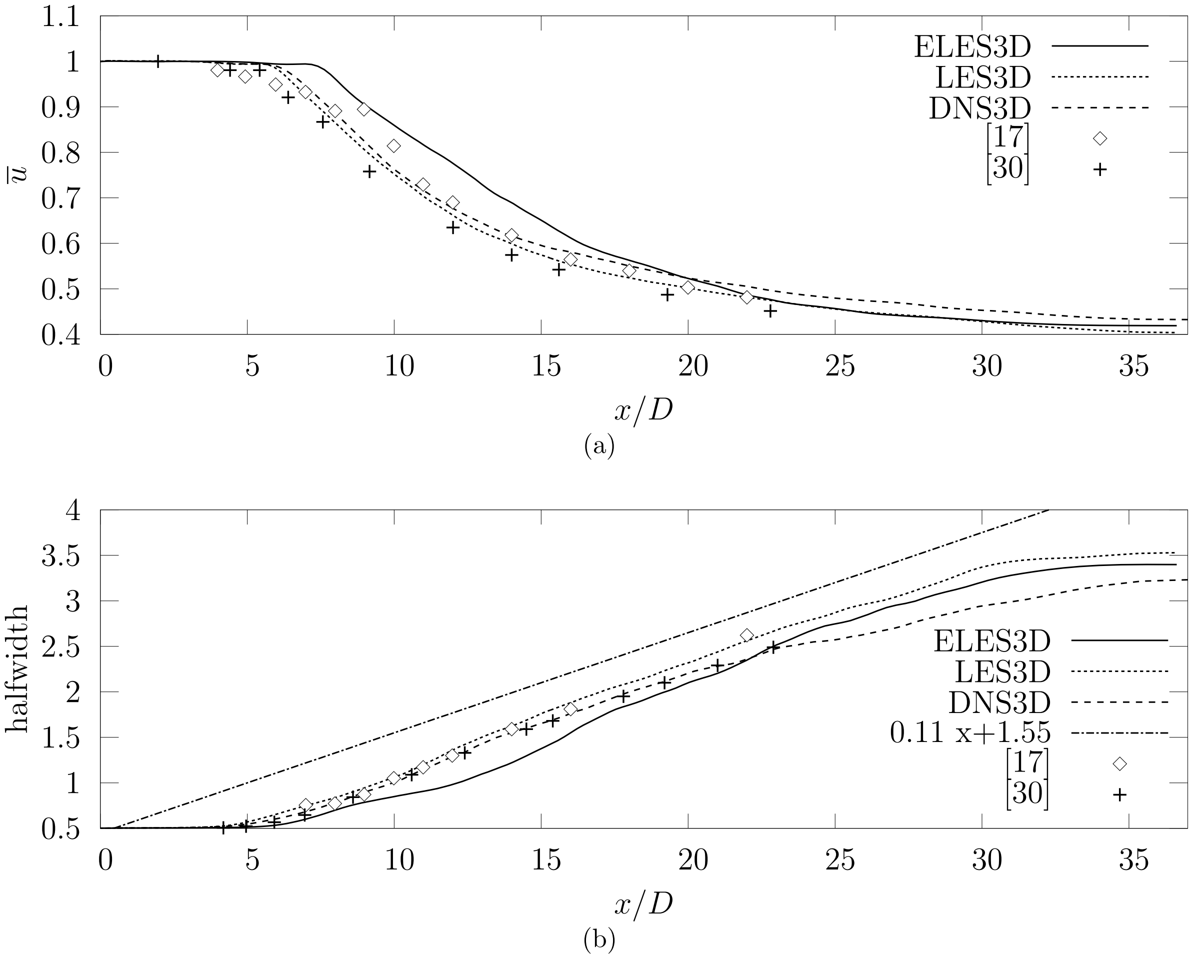


Figure 5.8: (a) $\overline{u}$ and (b) jet halfwidth for cases ELES3D, LES3D and DNS3D along the centerline of the jets. For reference also a line with a gradient of 0.11 is shown. For comparison results from other works are shown, too. The values from [17] and [30] have been shifted in streamwise direction.

5.4. Jet Statistics

To validate the jet simulations typical statistical values are compared with the literature in this section. The streamwise velocity along the centerline is shown in figure 5.8(a). It can be observed in 5.8(a) that the velocity decay starts later for case ELES3D. During the simulation difficulties have been encountered to trigger the transition to turbulence, which coincides with the beginning of the jet centerline velocity decay, as soon as in cases LES3D and DNS3D. This might be a consequence of the broader initial shearlayer width or the increased numerical dissipation due to the coarser resolution of case ELES3D. For comparison the streamwise velocity obtained from different experimental [17] and numerical [30] investigations are shown in figure 5.8(a), too, and a good agreement can be observed.

The jet halfwidth, defined as the distance from the centerline where the mean downstream velocity reaches half its maximum value

$$z_{1/2}(x) = |z_{\text{halfwidth}} - z_{\text{centerline}}| \quad \text{with} \quad \overline{u}(x, z_{\text{halfwidth}}) \overset{!}{=} \frac{1}{2}\overline{u}(x, z_{\text{centerline}}) \tag{5.16}$$

can be observed in figure 5.8(b). To estimate the growth rate of the halfwidth a line with a slope of 0.11 is shown, too. [105] gives a growth rate range of 0.1 to 0.11 and in [87] values between 0.092 and 0.18 are reported. Thus, the growth rate found in this work lies within the range found in the literature. As for the jet centerline velocity decay a good agreement with other works can be found.

Exploiting symmetry and assuming conservation of momentum a self similar profile for the mean streamwise velocity can be derived for a plane jet [73, section 5.4.2], which is given by

$$\frac{\overline{u}(x,z)}{\Delta U_c(x)} = \operatorname{sech}\left(\frac{1}{2}\log\left(\left(1+\sqrt{2}\right)^2\right)\frac{z}{z_{1/2}(x)}\right)^2. \tag{5.17}$$

The spanwise profiles of the streamwise velocity are shown at different locations in downstream direction for the different grids in figure 5.9 and are compared to the theoretical solution. At $x = 4D$ the flow isn't fully developed. Thus, the hyperbolic tangent profile enforced at the inflow is still recognizable. However, further downstream at $x = 8D$, $x = 12D$ and $x = 20$ the profiles agree well amongst each other and in comparison to the solution in equation (5.17). It can be observed that the velocity decay in spanwise direction is more rapid for the computed solutions in the tail of the profiles compared to the theoretical solution. This observation is in accordance with experiments [73, figure 5.19].

Figure 5.10 shows the root of the averaged Reynolds stresses $R_{u_i u_i} = \overline{\widehat{u_i}\widehat{u_i}} = \overline{u_i u_i} - \overline{\rho u_i}\;\overline{\rho u_i}$ normalized with the jet centerline velocity for the stream- and shearwise velocities along the jet centerline. For validation these values are compared with results form other experimental and numerical works. The same works served as a validation in [87, figure 11]. The values found for the Reynolds stresses in this work exceed the reference stresses. However, the reference values were obtained at slightly higher Reynolds numbers ([87]: $Re = 3000$; [96]: $Re = 8300$; [97]: $Re = 8000$; [17]: $Re = 7620$) and at lower Mach number rendering the reference cases practically incompressible. Furthermore, the turbulence level shows a strong dependence on the inflow forcing [12, 15]. Thus, the agreement observed in figure 5.10 seems reasonable.

The pressure fluctuations in the farfield for the different grids are compared in figure 5.11. Case ELES3D and LES3D agree quite well. Interestingly, case DNS3D doesn't exhibit the distinct noise peak around $x \approx 27D$ observed in cases ELES3D and LES3D. However, variations in the noise level of several decibel due to small changes in the inflow boundary

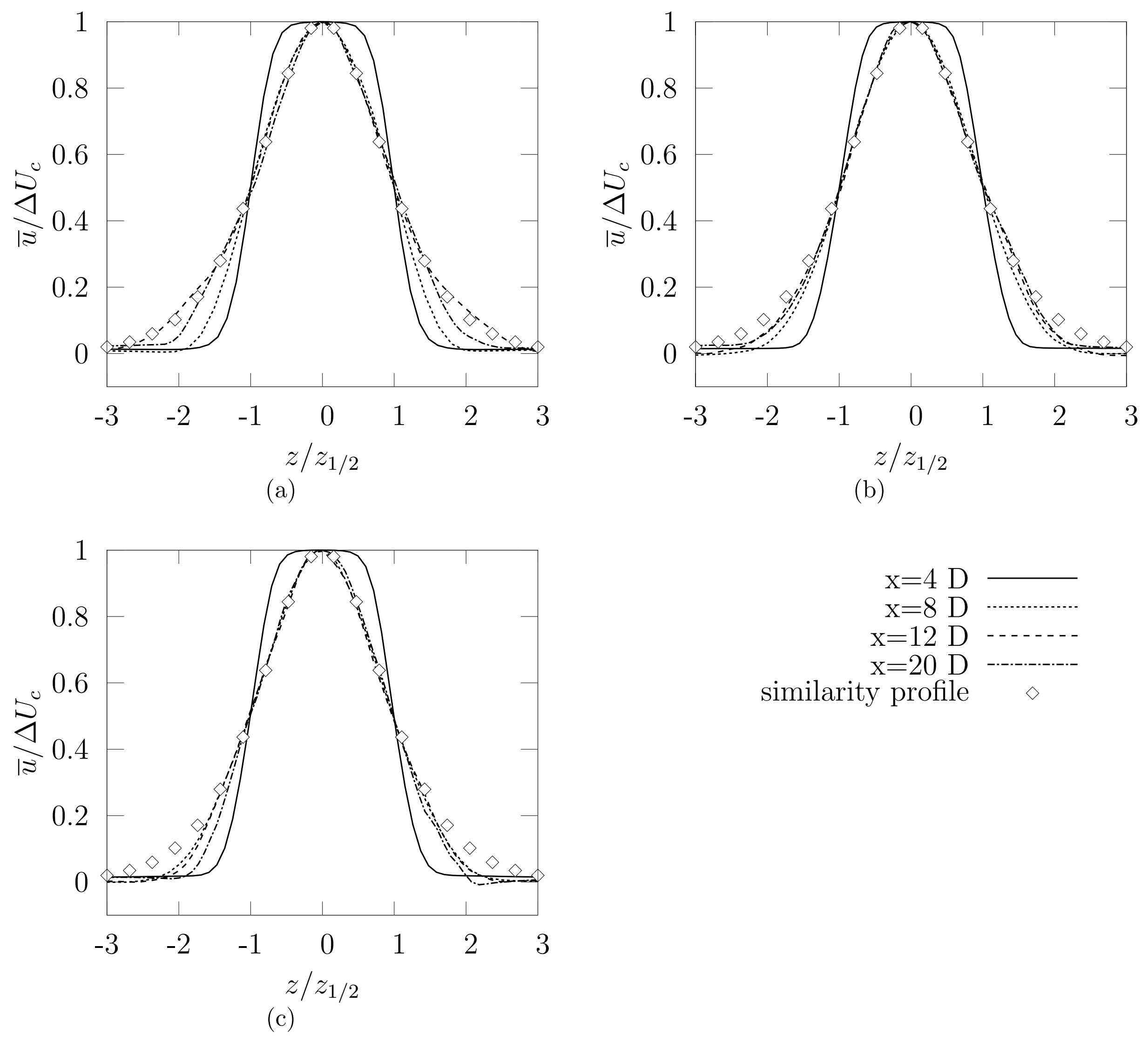
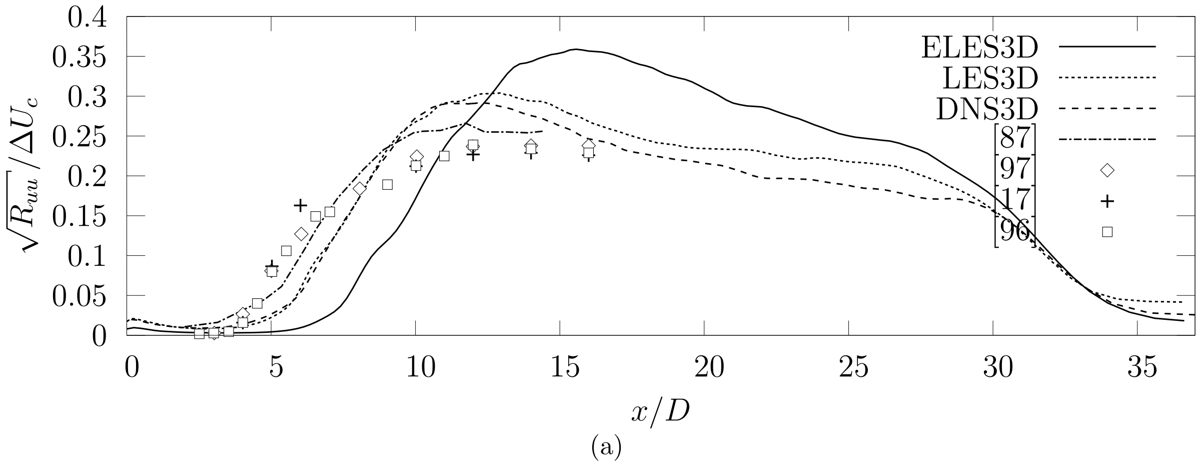
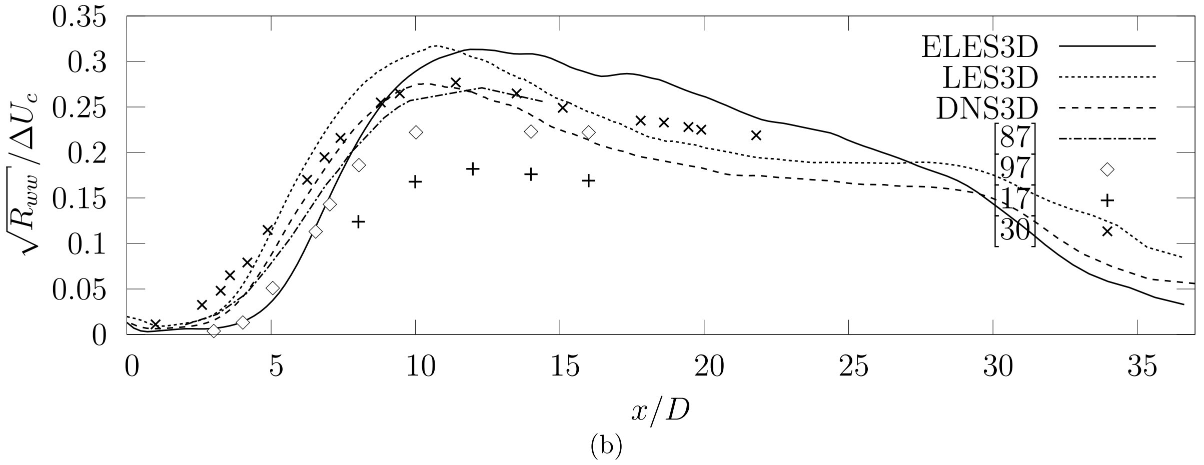


Figure 5.9: Streamwise velocity in spanwise direction at different coordinates for cases (a) ELES3D, (b) LES3D and (c) DNS3D. The profiles are normalized with the jet halfwidth and centerline velocity to illustrate the self similarity.

conditions are reported in the literature [9, 12]. As the cases in this work involve different ranges of scales due to the different resolutions a high agreement is not to be expected. A decrease of sound pressure level (SPL) with increasing resolution was also observed in [14].



(a)



(b)

Figure 5.10: Averaged Reynolds stresses (a) $\sqrt{R_{uu}}/\Delta U_c$ and (b) $\sqrt{R_{ww}}/\Delta U_c$ for cases ELES3D, LES3D and DNS3D along the centerline of the jets together with some experimental and numerical results for validation. The comparison values have been shifted in streamwise direction.

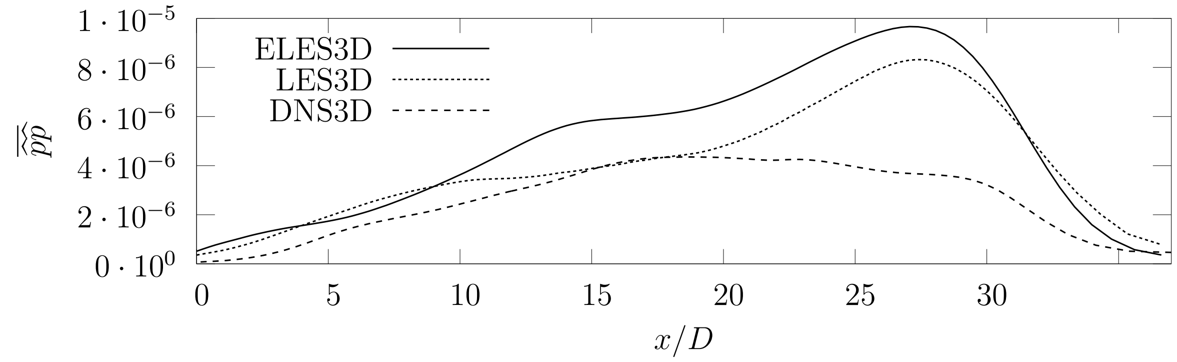


Figure 5.11: Pressure fluctuations $\overline{\widehat{p}\widehat{p}}$ for cases ELES3D, LES3D and DNS3D in the farfield of the jets. The $\bar{\cdot}$ denotes an average in time and in the periodic spanwise direction and a $\widehat{\cdot}$ denotes the fluctuating part.

6. Gradient Accuracy

6.1. Validation of Discrete Adjoint

To ensure the correct implementation of the discrete adjoint several tests have been performed to check the validity and correctness of the implementational approach of this work. Complex differentiation and operator overloading were used to check the correct implementation of the first and second order sensitivity equations as described in section 4.5. Computing the linearized RHS operator with randomly chosen flow variables and sensitivities the relative difference between the results of these methods is in the order of 10^{-16}, which is to be expected due to the floating point representation in memory. No additional tests were performed to ensure the correct linearization of the RK iteration as it is already linear.

With the sensitivity equations validated they can be used to validate the RHS operator of the adjoint equations using the identity $\mathbf{a}^T \left(\frac{\partial \mathbf{R}}{\partial \mathbf{\Phi}} \big|_{\mathbf{\Phi_s}} \right)^T \mathbf{b} = \mathbf{b}^T \left(\frac{\partial \mathbf{R}}{\partial \mathbf{\Phi}} \big|_{\mathbf{\Phi_s}} \right) \mathbf{a}$ with arbitrary vectors $\mathbf{a}$ and $\mathbf{b}$, where $\mathbf{R}$ is the discretized Navier-Stokes operator. In several tests done with randomly chosen vectors an agreement between LHS and RHS could be observed down to a precision in the order of 10^{-16}.

The complete implementation of the adjoint system (4.37)-(4.41) can be validated with the linear response of the cost functional $\frac{d\Im}{d\mathbf{g}}\mathbf{g}'$ with respect to some arbitrary perturbation $\mathbf{g}'$. This linear response can either be obtained from the solution of the sensitivity equations, or by the first order term of a Taylor series expansion:

$$\underbrace{\frac{\partial L}{\partial \mathbf{g}}}_{\text{gradient}} \mathbf{g}' = \frac{d\Im}{d\mathbf{g}}\mathbf{g}' = \frac{\partial \Im}{\partial \mathbf{\Phi}} \frac{d\mathbf{\Phi}}{d\mathbf{g}}\mathbf{g}' + \frac{\partial \Im}{\partial g}\mathbf{g}' = \frac{\partial \Im}{\partial \mathbf{\Phi}} \underbrace{\mathbf{\Phi}'}_{\text{sensitivity}} + \frac{\partial \Im}{\partial g}\mathbf{g}' \tag{6.1}$$

case	pos.	#RK	$\frac{\partial L}{\partial \mathbf{g}}\mathbf{g}'$	$\frac{\partial \Im}{\partial \mathbf{\Phi}}\mathbf{\Phi}' + \frac{\partial \Im}{\partial g}\mathbf{g}'$
DNS2D	100	10000	$-2.19978130134220 \cdot 10^{-4}$	$-2.19978130134229 \cdot 10^{-4}$
	900	10000	$1.44380102098536 \cdot 10^{-5}$	$1.44380102098535 \cdot 10^{-5}$
	1700	10000	$-1.17132271501490 \cdot 10^{-5}$	$-1.17132271501492 \cdot 10^{-5}$
ELES3D	100	2400	$5.37171660047461 \cdot 10^{-6}$	$5.37171660047416 \cdot 10^{-6}$
	200	2400	$-6.73097452538381 \cdot 10^{-6}$	$-6.73097452538368 \cdot 10^{-6}$
LES3D	100	6400	$2.25323809359032 \cdot 10^{-4}$	$2.25323809359047 \cdot 10^{-4}$
	200	6400	$-1.70187568261517 \cdot 10^{-4}$	$-1.70187568261540 \cdot 10^{-4}$
DNS3D	100	7000	$3.11011806150672 \cdot 10^{-4}$	$3.11011806150674 \cdot 10^{-4}$
	400	7000	$-2.85005052892766 \cdot 10^{-5}$	$-2.85005052892798 \cdot 10^{-5}$

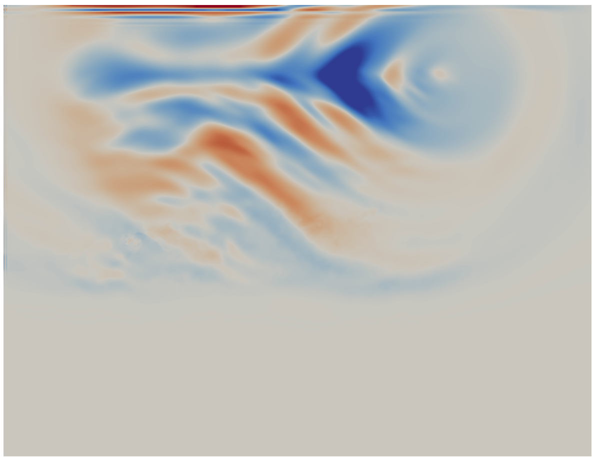
Table 6.1: Comparisons of LHS and RHS of equation (6.1) for different perturbations and the different jets simulated in this work. The "pos." gives the position of the perturbation in RK iterations from the beginning of the control interval. #RK gives the number RK iterations of the control interval. The maximal amplitude of the perturbations was 10^{-4} for all cases.

where L is the Lagrangian given in equation 4.34 and $\cdot'$ denotes variation with respect to the control. Several tests with small gaussian shaped perturbations with a finite temporal support were performed for the cases listed in table 5.2. Some of these tests are listed in table 6.1 and it becomes apparent that the LHS and RHS in equation (6.1) agree for at least 12 decimal digits. Note, that a high number of these validation tests isn't necessary as an agreement of 12 decimal digits by chance is exorbitantly unlikely. The full discrete adjoint over the complete time horizon is not accurate down to machine precision. As it was validated that the RHS is transposed accurately this is most likely due to round up errors during the RK iterations and the integration of the long time horizon. Nevertheless, due to the still excellent precision of the discrete adjoint it will be called exact in the following. Furthermore, the discrete adjoint will be used as a reference solution to validate the accuracy of other adjoint formulations. Note that exact is just meant in the sense that the implemented adjoint system is the exact adjoint system of the discretized system.

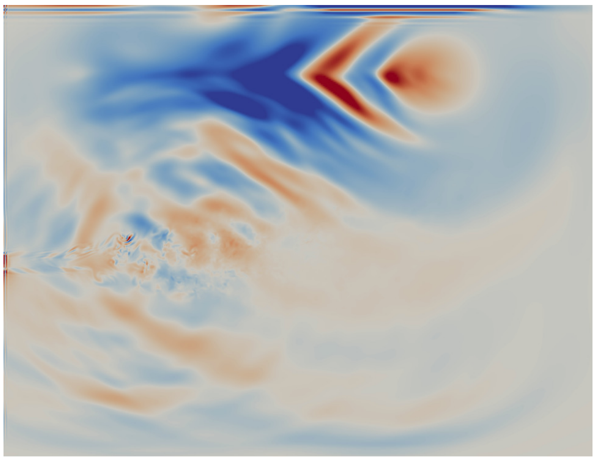
Given two arbitrary perturbations g' and g° the implementation of the Hessian computation can be tested by comparing the value of $\left.\frac{d^2\Im}{dg^2}\right|_g [g', g^\circ]$ obtained using either the sensitivity or adjoint equations. Furthermore, the symmetry of the Hessian $\left.\frac{d^2\Im}{dg^2}\right|_g [g', g^\circ] = \left.\frac{d^2\Im}{dg^2}\right|_g [g^\circ, g']$ was exploited to validate the accuracy of the adjoint solution. As for the linear terms an agreement of 12 decimal digits or more could be observed.

6.2. Appearance of Adjoint Solution

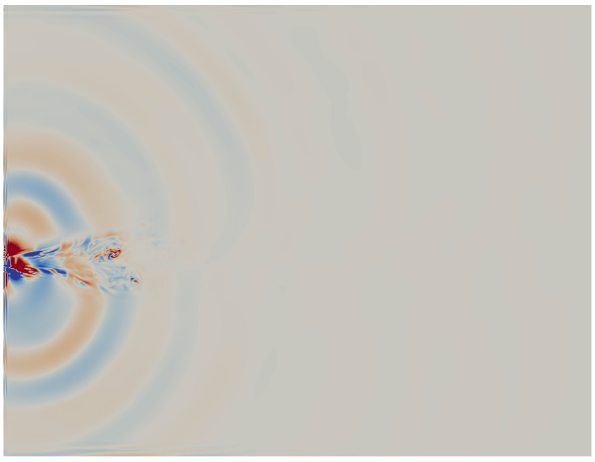
In order to describe the typical behavior of the adjoint solution, instantaneous solutions of the "adjoint pressure" (p^* in equation (4.86)) at different times are shown in figure 6.1 for case LES3D. The figures show the complete computational domain including the sponge



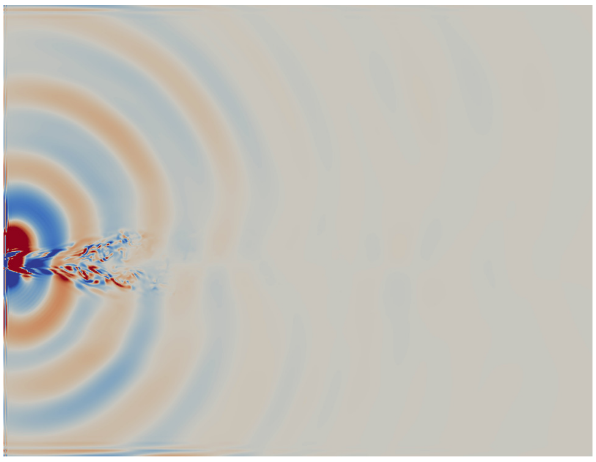
(a) t=121



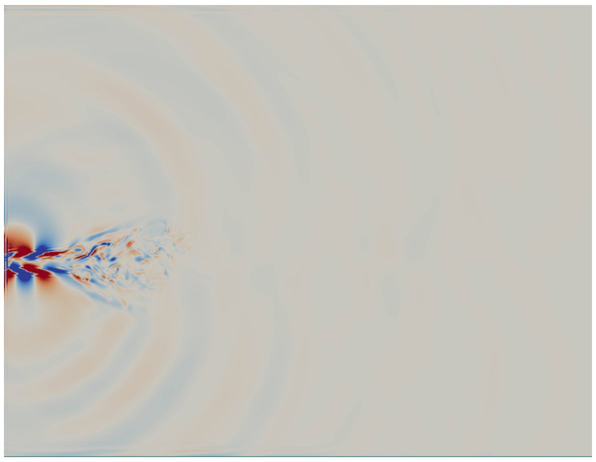
(b) t=113



(c) t=75



(d) t=12



(e) t=12 (sponged inflow)

Figure 6.1: Cross section of instantaneous discrete "adjoint pressure" at different times for case LES3D. The color scheme is identical in all cases, but (c) was scaled by a factor of 5×10^{-3}, (d) by $2 \cdot 10^{-5}$ and (e) by $1 \cdot 10^{-4}$. The inflow was sponged in (e).

zones. At an early stage of the adjoint solution (near the end of the optimization interval) the adjoint pressure is zero except near the area Ω, where the adjoint solution is perturbed due to the optimality condition. This is illustrated in figure 6.1(a). These perturbations radiate from region Ω with the speed of sound and begin to interact with the turbulent structures of the jet, as can be seen in figure 6.1(b). At even earlier times a part of the adjoint solution that can be associated with the turbulent region and especially the initial shear layers of the jet begins to grow fast in amplitude and becomes the dominant part of the adjoint.

At times $tU_j/c_j \approx 75$ sound wave like structures begin to emerge from the jet inflow. These "adjoint sound waves" indicate that it is possible to control the jet by sending sound waves towards the inflow of the jet and thereby altering the inflow conditions. For this work NBC were implemented. Thus, waves leaving the computational domain should cross the boundaries unperturbed and not influence the flow solution. However, only the normal part of the waves is transmitted, the tangential part is still active, leading to waves travelling parallel to the boundaries which need to be damped. Apparently, the imperfection of the CBC still allows for some control of the inflow from the inner region of the computation. The interpretation that waves leaving the computational domain at oblique angles do not leave the domain unperturbed and allow for a control of the flow solution is also supported by the observation that the "adjoint sound waves" emerging near the inflow have higher amplitudes at oblique angles. This indicates that waves at oblique angles are more efficient for the control of the inflow boundary.

Although reasonable from a numerical point of view, the possibility to control the inflow from within the computational domain is rather a numerical artefact due to the imperfection of the CBC. Thus, this kind of control mechanism might be unpleasant from a physical point of view. As already mentioned in section 5.3.1 different types of sponge zones have been used near the inflow. The solutions in figures 6.1(a)-6.1(d) are obtained from a computation using a weak inflow sponge. The reflections at the inflow can be reduced by using a strong sponge at the inflow. An adjoint solution for this case is shown in figure 6.1(e). This way, the influence of the reflections on the gradient accuracy can be investigated and the physical meaning of the adjoint is increased.

A close look at figure 6.1(d) reveals grid to grid oscillations at the boundaries. Similar observations have been made in [20]. This aliasing is a consequence of the transposition of the finite difference derivative and filter operators, which are asymmetric at the boundaries. It does, however, not lead to numerical instability. The same phenomenon cannot be observed in the continuous adjoint solution. The instantaneous adjoint solutions shown in figure 6.1 are obtained solving the discrete adjoint equations. However, except for the grid oscillations at the boundaries the discrete and continuous adjoint exhibit the same qualitative behavior and the description given in this section remains valid for the continuous adjoint, too.

Case	RK	RHS	f_{save}	save-type	boundary
ContNoBC	continuous	continuous	2	double	non
ContSpng	continuous	continuous	2	double	sponge
ContSgl	continuous	continuous	4	single	non-reflecting
ContF2	continuous	continuous	2	double	non-reflecting
ContF4	continuous	continuous	4	double	non-reflecting
ContF8	continuous	continuous	8	double	non-reflecting
ContF16	continuous	continuous	16	double	non-reflecting
Mixed	discrete	continuous	recompute	double	non-reflecting
DiscRef	discrete	discrete	recompute	double	discrete
DiscSgl	discrete	discrete	4/recompute	single	discrete
DiscF2	discrete	discrete	2	double	discrete
DiscF4	discrete	discrete	4	double	discrete
DiscF8	discrete	discrete	8	double	discrete
DiscF16	discrete	discrete	16	double	discrete

Table 6.2: List of the different cases considered for testing of the gradient accuracy for case LES3D. RK: formulation used for the RK iteration. RHS: formulation used for the RHS. f_{save}: number of RK iterations after which the flow field was saved. Save-type: single or double precision. Boundary: details about boundary treatment.

6.3. Continuous Adjoint

The knowledge of the exact discrete adjoint solution is used to test the accuracy of several different adjoint approaches in this and the following sections. The different approaches investigated are listed in table 6.2 and will be explained in detail below. The control simulations were carried out for case LES3D with a time horizon of 6400 RK iterations, giving a simulation length of $134D/U_j$. For the rest of this chapter a volume temperature forcing according to equation (5.5) is applied as control. However, as the different adjoint quantities are related to each other it is very unlikely that the inaccuracies grow differently for the different adjoint quantities. Thus, we expect the results of this chapter to be independent of the specific control type.

To quantify the accuracy of the different approaches two measures are defined. The correlation coefficient between the exact gradient (obtained from the discrete adjoint) $\mathbf{x}$ and the inexact gradient $\mathbf{a}$ at time t is defined as

$$\mathrm{corr}(\mathbf{x}, \mathbf{a}) = \frac{x_i a_i}{\sqrt{x_j x_j a_k a_k}}. \tag{6.2}$$

Note, that this correlation coefficient also corresponds to the cosine of the angle between

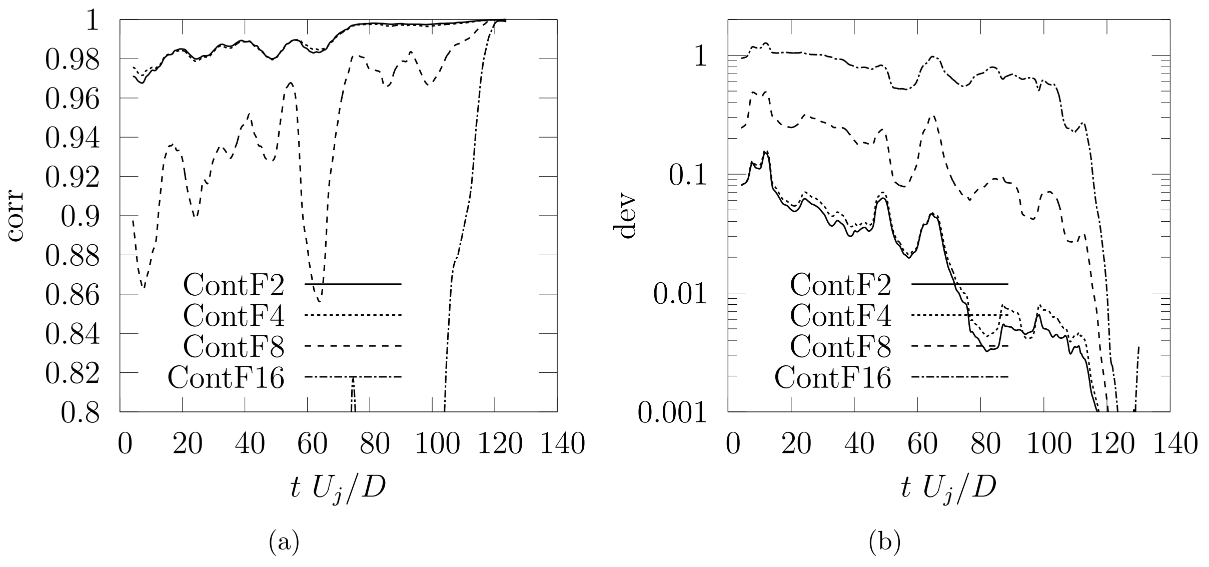


Figure 6.2: The correlation/deviation of the continuous adjoint gradient for different saving frequencies with/from the exact gradient for case LES3D. The gradient isn't accurate for saving frequencies of 8 and higher.

$\mathbf{x}$ and $\mathbf{a}$. The relative deviation of the two gradients is calculated with

$$\text{dev}(\mathbf{x}, \mathbf{a}) = \frac{||\mathbf{x} - \mathbf{a}||_2}{||\mathbf{x}||_2}. \tag{6.3}$$

To avoid spurious correlations caused by the window function s_T only gradient values with a function value of s_T near to one were considered. Additionally, the signals were averaged over 400 time steps, corresponding to a time interval of $8.4D/U_j$, to give a measure of the average deviation.

The flow fields, used for reconstruction in the adjoint equations, have been saved every 2nd, 4th, 8th and 16th RK iteration for the continuous adjoint approach. These cases are denoted as ContF2, ContF4, ContF8 and ContF16, respectively. The correlation coefficients and deviations for these cases are shown in figure 6.2 and indicate that case ContF4 is almost as accurate as case ContF2. On the other hand, case ContF8 shows considerably higher deviations from the exact solution and case ContF16 is even more inaccurate. This behavior can be explained by looking at figure 6.3, which shows the autocorrelations and powerspectra of the primitive flow variables ρ, u, v, w and T. The statistics were taken from the flow fields saved for the adjoint computation at a point on the centerline in the turbulent region of the jet. In this region the shortest correlation length and highest frequencies appear in the flow solution. The five primitive flow variables have been chosen as these quantities were saved/loaded during the flow/adjoint solution. A significant decrease of the autocorrelation can be observed when storing only every eighth RK iteration and higher examining figure 6.3(a). The reason for the drop in accuracy can be observed even clearer by looking at the powerspectra in figure 6.3(b). For cases ContF2 and ContF4

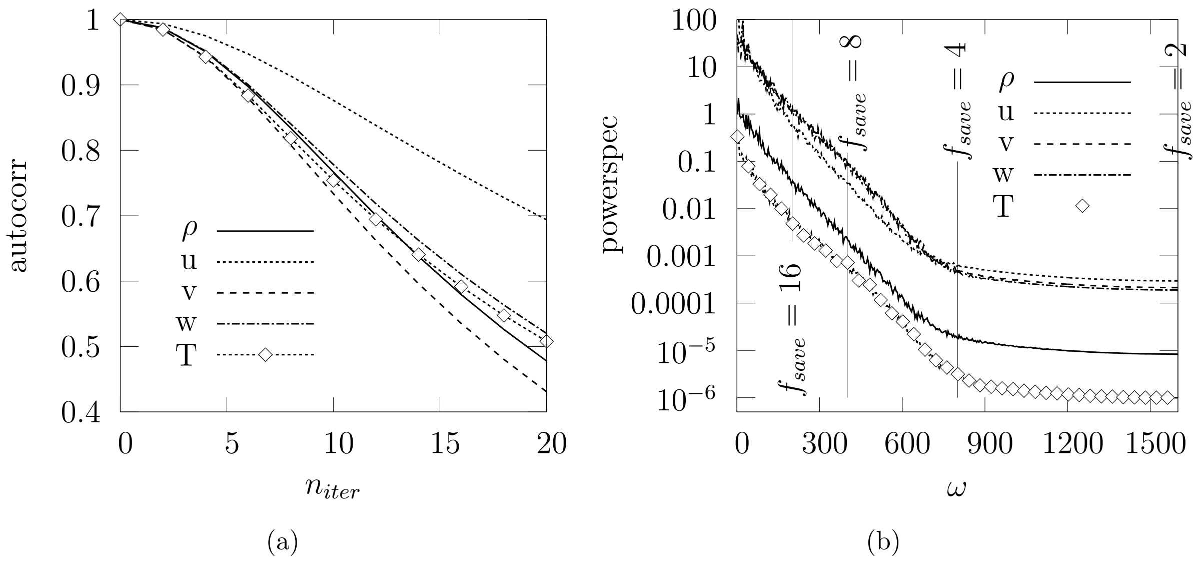


Figure 6.3: (a) Correlation and (b) power spectrum of density, velocities and temperature at the jet centerline in the fully turbulent part of the jet (x=13, z=12). The horizontal lines in (b) correspond to the Nyquist frequency for the different saving frequencies. An information loss of the time signal becomes evident for saving frequencies of 8 and higher.

the truncated modes seem to contain mainly white noise. The same can not be stated for cases ContF8 and ContF16. However, the neglected modes contain only about 0.1% of the energy of the total powerspectrum in case ContF8. This indicates the necessity of a very accurate flow reconstruction to obtain accurate gradients.

An accuracy drop becomes apparent at $tU_j/c_j \approx 75$ in figure 6.2. This moment corresponds to the situation in figure 6.1(c), where "sound waves" emerge from the inflow boundary in the discrete adjoint solution. As already discussed in section 4.7.1.2, the boundary conditions were constructed to be non-reflective and do not represent the exact continuous adjoint of the CBC used in the flow simulation (see also [59]). Accordingly, it is reasonable to assume that the errors introduced with the continuous adjoint approach are higher at the boundaries than in the inner region of the computational domain. Thus, it is reasonable to connect the accuracy drop to the appearance of the "adjoint sound waves" emerging from the boundary.

6.4. Boundary Conditions

The BC of the continuous adjoint formulation were altered to examine their influence on the accuracy of the adjoint. The correlation coefficients and deviations for these cases are shown in figure 6.4. To demonstrate that proper boundary conditions for the continuous adjoint are necessary, no boundary conditions were used in case ContNoBC, meaning that the same expression was used at the inflow boundary than in the inner regions of the computational

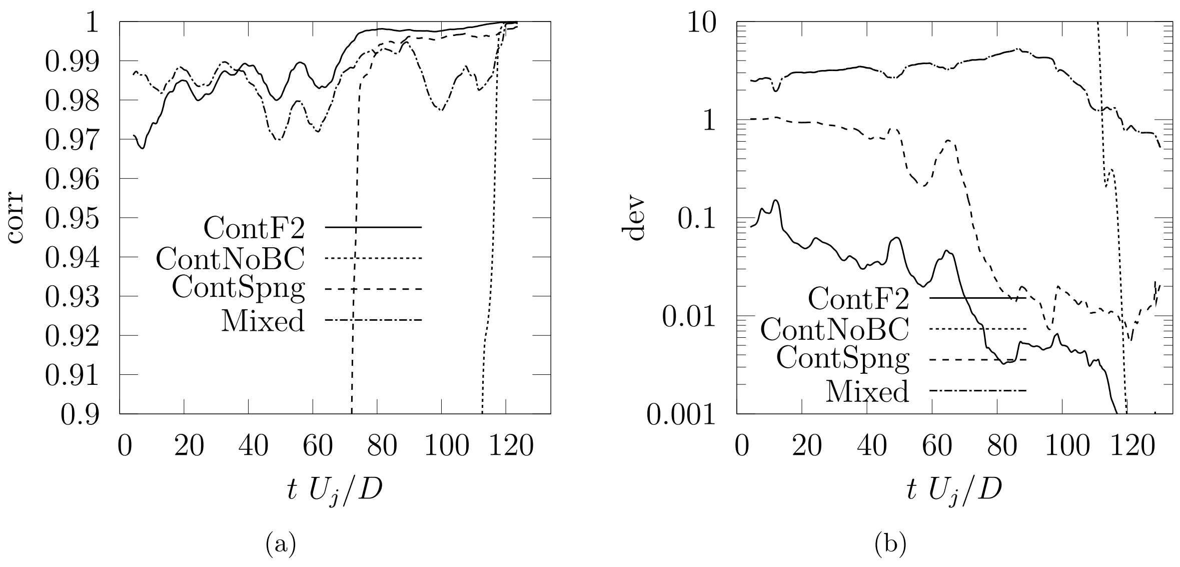


Figure 6.4: The correlation/deviation of the continuous adjoint gradient for different formulations of boundary conditions with/from the exact gradient for case LES3D. As discussed in the text, it is seen that proper adjoint boundary conditions are important.

domain. As is to be expected, the numerical scheme turns out to be unstable without boundary conditions. The computation breaks down at $tU_j/c_j \approx 115$, which corresponds to the instant when the adjoint solution reaches the inflow boundary (see figure 6.1(b)).

In order to stabilize case ContNoBC a strong sponge region is added at the jet inflow for the adjoint solution in case ContSpng. This sponge is not added to the primal flow solution and further inconsistencies between primal and ajdoint solution are introduced. However, as the adjoint solution convects towards the inflow the hope is that the inflow sponge in the adjoint solution has only a minor effect for the adjoint solution further upstream. Furthermore, using a sponge can serve as a valid boundary condition for compressible flows as was shown by [8, 58], for example. The discussion in the literature and the above arguments indicate that using a sponge zone is a valid boundary condition. To no surprise the inflow sponge is able to stabilize the computation of the adjoint. However, the accuracy is acceptable for times $tU_j/D \gtrsim 75$, only. Thereafter, the correlation quickly reaches values around zero (not shown) and the amplitude of the adjoint is largely underestimated. One explanation for this behavior is that from this point on reflections begin to emerge from the inflow in the adjoint solution (see figure 6.1(c)). These reflections are suppressed by the inflow sponging of the adjoint and case ContSpng deteriorates from the solution of the discrete adjoint.

Next, case ContF2 is repeated with a strong inflow sponge consistently applied for the primal and adjoint flow solution. This way the reflections at the inflow are suppressed (see figure 6.1(e)) without introducing further inconsistencies in the adjoint solution as in the previous case. The reasoning is that the sponge forcing decreases the sensitivity near

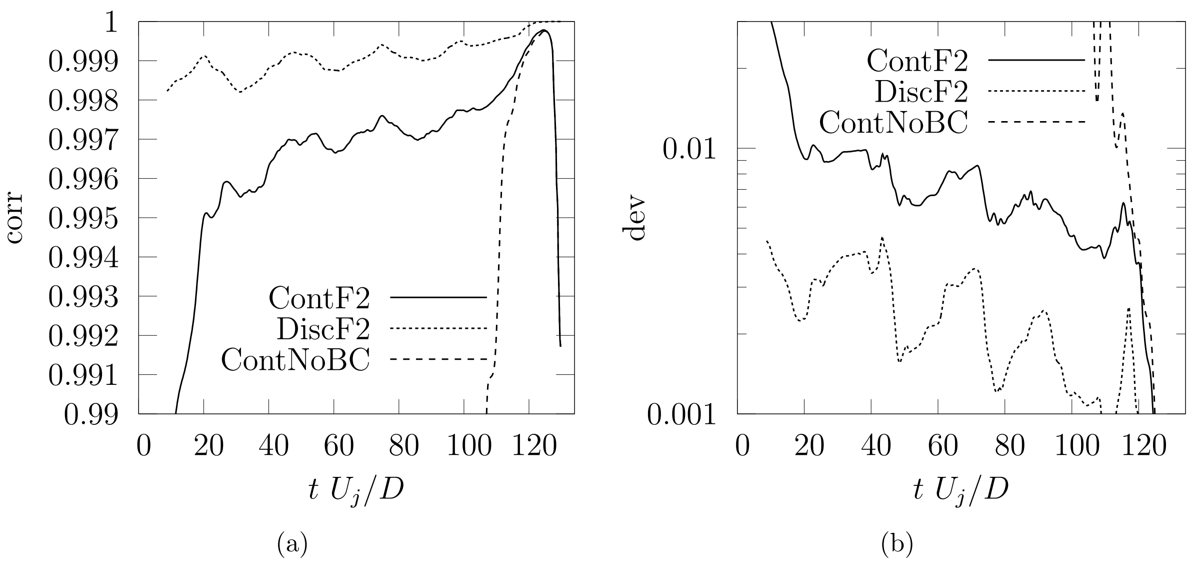


Figure 6.5: The correlation/deviation of the continuous and discrete adjoint gradient for case LES3D with a strong inflow sponge. Although an inflow sponge region is added to the adjoint equations, the gradient becomes inaccurate without non-reflective boundary conditions.

the inflow and the boundary can not be controlled from inside the computational domain any more. As a consequence the inconsistencies observed at the boundary should become less relevant. Consequently, figure 6.5, showing the correlation coefficient and deviation of the continuous adjoint subject to a strong sponge region in comparison to the discrete adjoint solution, depicts that the gradient accuracy is increased (compare with case ContF2 in figure 6.4). Especially, the accuracy drop mentioned before at $tU_j/c_j \approx 75$ cannot be observed anymore. Figure 6.5 also depicts the accuracy of case DiscF2, a discrete adjoint formulation without recomputation explained in more detail in section 6.5. Although the errors connected to the boundary treatment are minimized by the sponge zone case DiscF2 is more accurate than ContF2. Using the same RHS formulation for the inner and boundary grid points as in case ContNoBC the correlation of this adjoint quickly deteriorates from the exact solution. This observation may not necessarily be true for longer or stronger sponge regions. Nevertheless, it demonstrates the importance of proper adjoint boundary conditions for an efficient computation of the adjoint solution.

To summarize, the importance of proper boundary conditions were demonstrated and sponge zones do not serve as a valid adjoint boundary condition. The boundary conditions used in this work for the continuous adjoint proved to be able to give accurate gradient information. Nevertheless, additional inaccuracies are introduced in the continuous adjoint at the boundaries. These inaccuracies can be reduced by adding sponge zones, although this sponging has to be done consistently in the primal and adjoint solution.

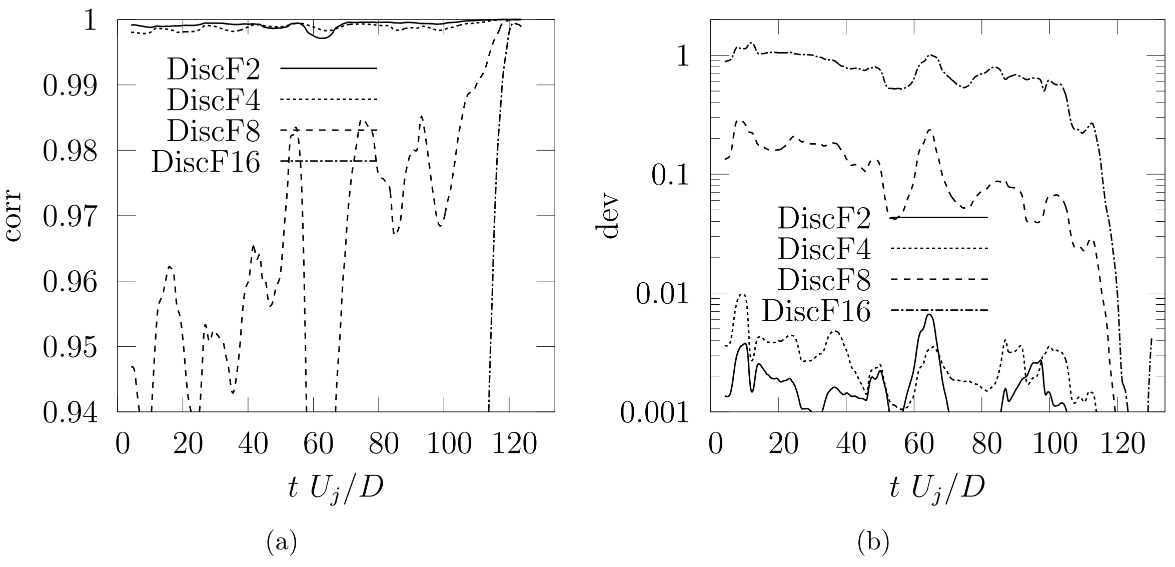


Figure 6.6: The correlation/deviation of the discrete adjoint gradient without recomputation of flow fields for different saving frequencies with/from the exact gradient for case LES3D. As in the continuous case (see figure 6.2) the gradient isn't accurate for saving frequencies of 8 and higher.

6.5. Discrete Adjoint

Computing the discrete adjoint requires the knowledge of the complete flow field at every RK substep. Due to memory restrictions and the backward in time integration of the adjoint equations this usually necessitates frequent recomputations of the primal flow solution for time dependent problems. This recomputation increases the computational effort and might be cumbersome from an implementational point of view.

The discrete adjoint formulation was used in cases DiscF2, DiscF4, DiscF8 and DiscF16. However, instead of recomputing the necessary flow fields they are saved and loaded every second, forth, eighth and 16^{th} RK integration and the flow fields at time steps within each storage interval are reconstructed using a third order accurate interpolation scheme, where the interpolation coefficients are calculated according to equation (4.5). Note, that by skipping the recomputation of the (numerically) exact flow fields the computational expense of the discrete adjoint is decreased. Figure 6.6 depicts the accuracy of these approaches. A high accuracy can be observed for cases DiscF2 and DiscF4. Especially, a comparison with figure 6.2 reveals that the cases DiscF2 and DiscF4 are more accurate than case ContF2 and ContF4. Furthermore, the accuracy drop observed for case ContF2 and ContF4 at $tU_j/D \approx 75$ can not be found for the cases using the discrete adjoint formulation. The finding that case DiscF2 is more accurate than case ContF2 also holds for the jet with a strong inflow sponging, as can be seen in figure 6.5. The accuracy drops significantly

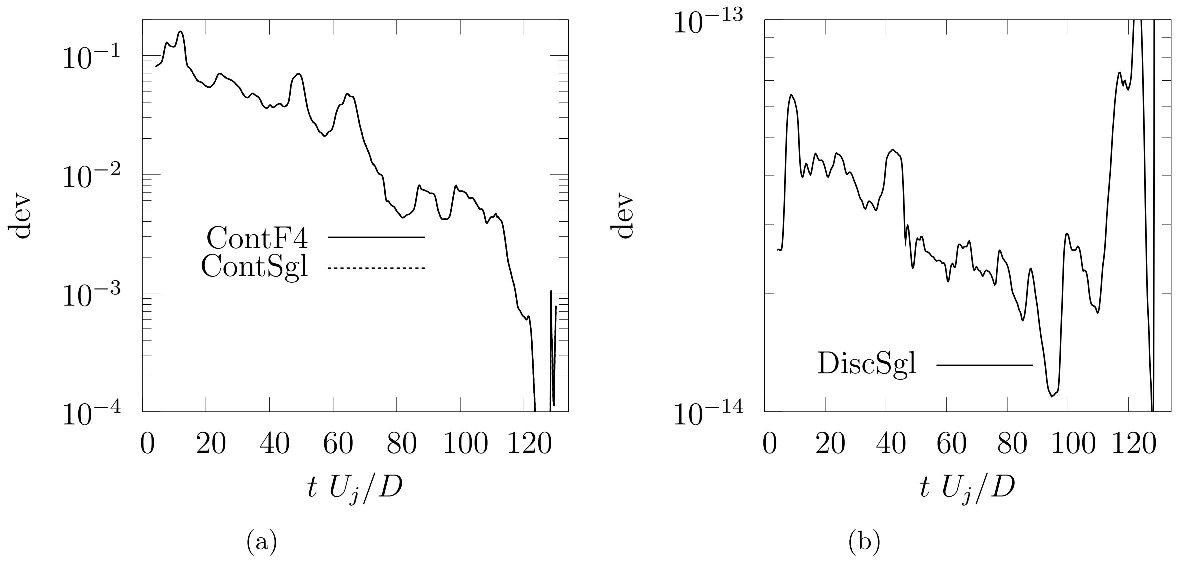


Figure 6.7: Deviation for cases (a) ContSgl and (b) DiscSgl.

for case DiscF8 and even more for case DiscF16. This is analogous to the behavior of the corresponding continuous adjoint cases and can again be explained with the decreased flow field reconstruction precision for decreased saving frequencies. Nevertheless, the case DiscF8 is still more accurate than ContF8.

A very simple strategy that reduces the hard disk I/O by a factor of two is to save the flow fields in single precision, albeit being calculated in double precision. The validity of this approach is tested for the discrete and continuous adjoint with the cases DiscSgl and ContSgl and the deviations for these cases are shown in figure 6.7. As expected, the inaccuracies introduced by the continuous adjoint approach outweigh the error introduced by saving in single precision. In fact, the deviations for cases ContSgl and ContF4 are indistinguishable as can be seen in figure 6.7(a). A drop in accuracy could be observed for case DiscSgl (figure 6.7(b). Nevertheless, cases DiscSgl and DiscRef still agreed up to six decimal digits (estimated by the square root of the deviation) and thus saving in single precision is a reasonable approximation for some applications.

Case Mixed constitutes a mixture of the continuous and discrete adjoint approach. The idea is to use the exactly transposed RK integration in equations (4.37)-(4.41). The computation of the term $\left(\frac{\partial R}{\partial \mathbf{\Phi}}\Big|_{\mathbf{\Phi_s}}\right)^T$ in equation (4.39), however, is avoided by substituting it with its continuous formulation in equation (4.84)-(4.86) and the boundary conditions of section 4.7.1.2. This approach might be interesting from a practical point of view, as the implementation of the computation of the linearized and transposed Navier-Stokes operator can be implementationally challenging. Examining figure 6.4, a high correlation with the exact gradient can be observed for case Mixed. The amplitude, however, is largely overestimated leading to a high deviation. It can be seen in figure 6.4 that an essential advantage of case Mixed compared to case ContF2 can not be observed, leading to the conclusion that no accuracy gain can be obtained by using the discrete adjoint RK integration in conjunction with the continuous RHS.

7. Noise Optimization

This chapter will demonstrate the successful application of the adjoint approach for noise optimization. Throughout this chapter the heat forcing according to equation (5.5) will be used as control mechanism. The choice of the control type can have an influence on the performance of the optimization [103, 47]. To illustrate this, figure 7.1 shows optimizations of case DNS2D using different control types as given in equations (5.3)-(5.5). More details about these optimizations can also be found in [47]. All control types successfully reduce the noise emission, although slight differences in the efficiency can be observed with the streamwise momentum forcing being the most efficient choice. However, in this work temperature forcing was chosen as control mechanism as this forcing was successfully applied in previous works [103, 50, 59]. Furthermore, of the controls given in equations (5.3)-(5.5) temperature control might be the one closest to experiments, as a local heat source could e.g. be applied using plasma actuators [79].

7.1. Comparison of Optimization Schemes

In this section the ability of the different optimization schemes introduced in chapter 2 to reduce noise in presence of a high dimensional control space are investigated. The minimizations in this section are carried out for the case ELES3D on a control horizon of $\Delta t U_j/D = 70$ (2400 RK iterations). This case was chosen as it contains the turbulent three dimensional dynamics of the Navier Stokes equations, but its rather low resolution enables one to perform a lot of optimization iterations. When comparing different optimization schemes a common measure for efficiency is the required number of cost function evaluations till a local minimum is reached, e.g. identified by observation of the norm of the gradient. Figure 7.2 shows exemplarily the L_2 norm of the gradient over the number of optimization iterations for a LBFGS optimization. It can be observed that except for a short initial period aside from statistical fluctuations no gradient norm reduction can be observed. This leads to the conclusion that, e.g. due to the high dimensionality of the control space or the complex structure of the cost functional, no local minimum has been reached.

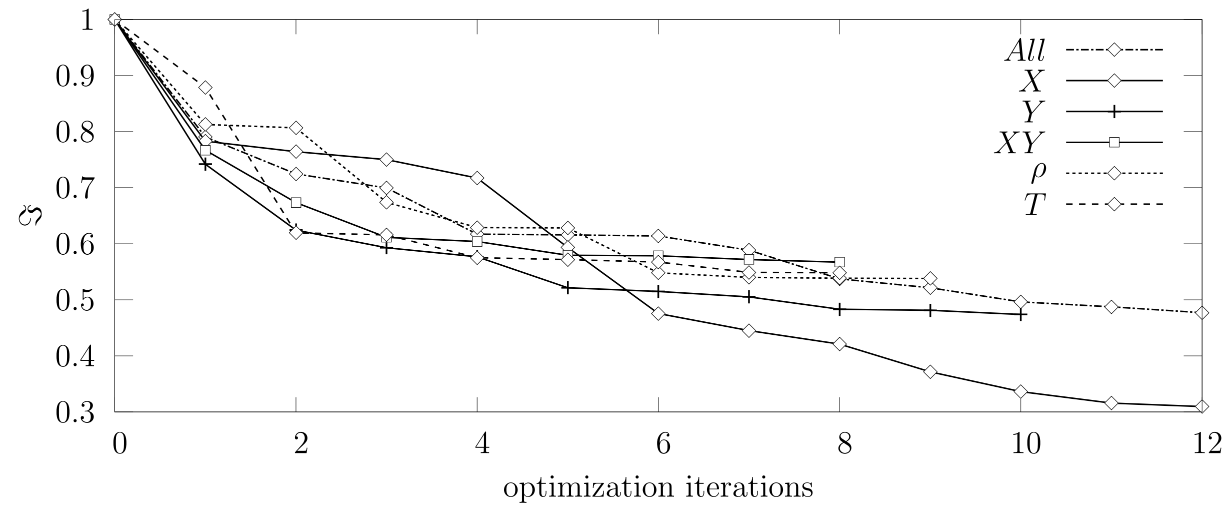


Figure 7.1: Cost functional over CG-iterations for case DNS2D with different control types performed in [47]. The control consisted of a volume forcing of density (ρ), streamwise momentum (X), shearwise momentum (Y), stream- and shearwise momentum (XY), temperature (T) and all of the above simultaneously (All). The cost function values are normalized with their initial value without control.

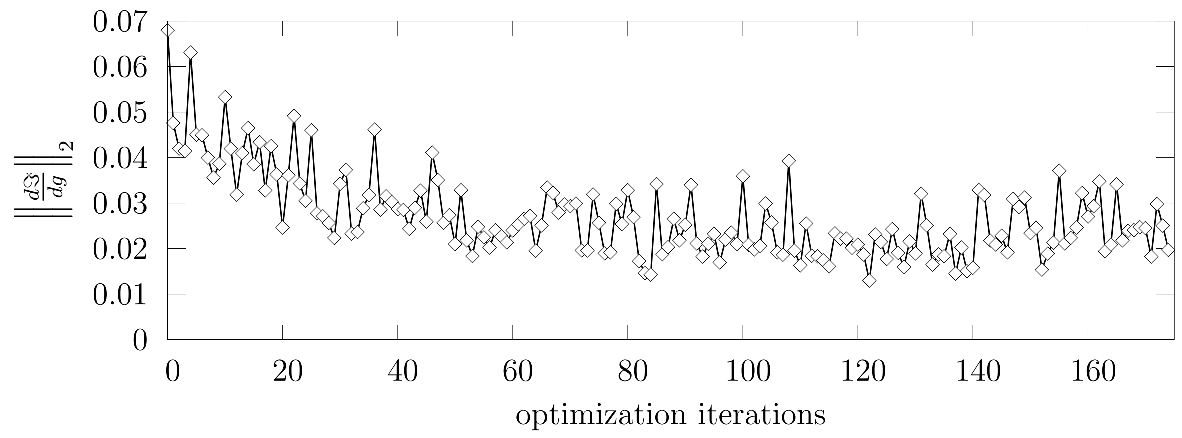
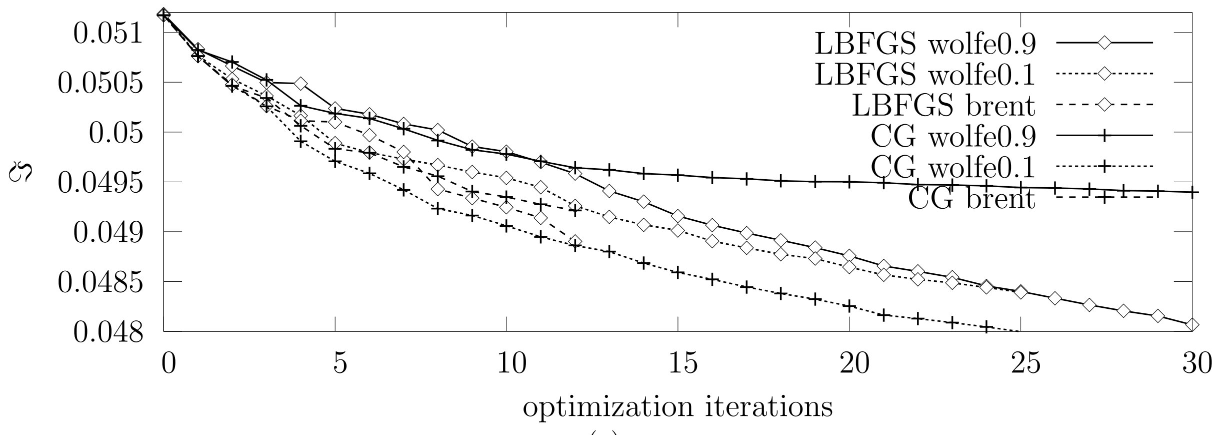


Figure 7.2: Norm of gradient versus iteration number for a LBFGS Wolfe09 optimization with case ELES3D. A persistent reduction of the gradient norm can not be observed.

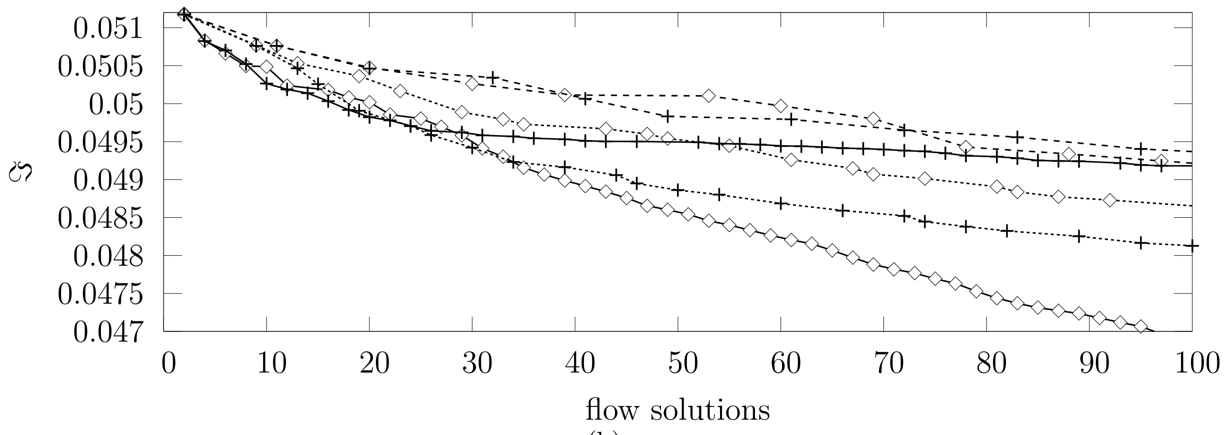
Thus, the optimizations are aborted after an arbitrary number of iterations and the cost functional reduction serves as a performance index.

7.1.1. Gradient Based Optimization Schemes

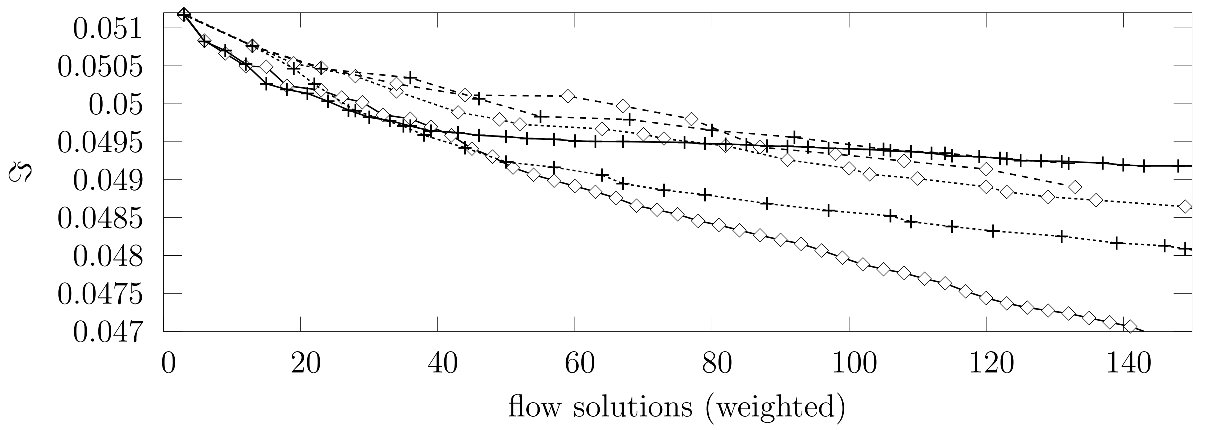
In this section the efficiency of the gradient based CG and LBFGS optimization schemes introduced in section 2.2.2 and 2.2.3.1 is investigated. As line search algorithm the Brent



(a)



(b)



(c)

Figure 7.3: Comparison of performance of LBFGS and CG optimization schemes for case ELES3D using Brent and Wolfe line searches.

line minimization of section 2.1.2 and the Wolfe line search of section 2.1.3 were used. The Brent line minimization was performed until the condition $\frac{\alpha_u - \alpha_l}{2\alpha} < 0.1$ holds, where α_u and α_l are the upper and lower estimates of the exact line minimization step lengths and α is the step length with the highest cost functional reduction found so far. The constant c_2 used in equation 2.6 for the curvature condition was chosen as $c_2 = 0.9$ or $c_2 = 0.1$. This line searches will be referred to as Brent01, Wolfe09 and Wolfe01, respectively. The specific cost functional reduction depends on the initial conditions and the choices of the optimization parameters (e.g. the constant c_2, or the control interval length). Thus, it would be necessary to test the optimization schemes with several different initial conditions to obtain a detailed picture of the optimization efficiency. However, due to the computational expense, one optimization with the same initial condition for all cases, was performed, only. Nevertheless, because of the rather high number of optimization iterations (about 100) the results of this section should still be sufficient to give an impression of the optimization efficiencies of the different schemes.

For the cases mentioned above the cost functional over optimization iterations is shown in figure 7.3(a). As is to be expected for both optimization schemes, the cost functional decreases faster for the more accurate line minimizations Brent01 and Wolfe01 compared to the less accurate Wolfe09 line search. For the CG optimization the Wolfe09 line search performs very poor, while on the other hand the CG optimization works best with the Wolfe01 line search. This can be explained by recalling that the CG scheme requires that the gradient is perpendicular to the prior search direction. This assumption is fulfilled only at local minima of the cost functional in the search direction, explaining the need for an accurate line search in conjunction with CG method. The Wolfe line search is constructed to identify a step length where the cost functional is flat in the search direction. This coincides with finding step lengths where the gradient is almost perpendicular to the search direction. This possibly explains why the Wolfe01 line search performs better than the Brent01 line search for the CG method.

Figure 7.3(a) might indicate that the CG-Wolfe01 or LBFGS-Brent01 optimization schemes are the most efficient ones. However, the more restrictive the criteria for the acceptance of a step length the more trial flow evaluations are necessary to find an appropriate step length. Thus, the cost functional reduction per optimization iteration is not a good measure if one is interested in optimization efficiency in terms of computational time. Therefore, the cost functional is plotted over the number of flow solutions in figure 7.3(b). Figure 7.3(c) is the same, but here an adjoint solution was weighted with a factor of 2.5. This factor corresponds roughly to the ratio

$$\frac{\text{computational time discrete adjoint solution}}{\text{computational time primal flow solution}}.$$

In accordance with previous reasoning the case Wolfe01 performs best for the CG scheme in terms of computational time. For the LBFGS scheme the Wolfe09 line search is more efficient than the Wolfe01 or Brent01 line searches, indicating that an accurate line search doesn't pay off for this optimization scheme. Moreover, case LBFGS-Wolfe09 seems to outperform all other schemes due to its inexpensive line search routine.

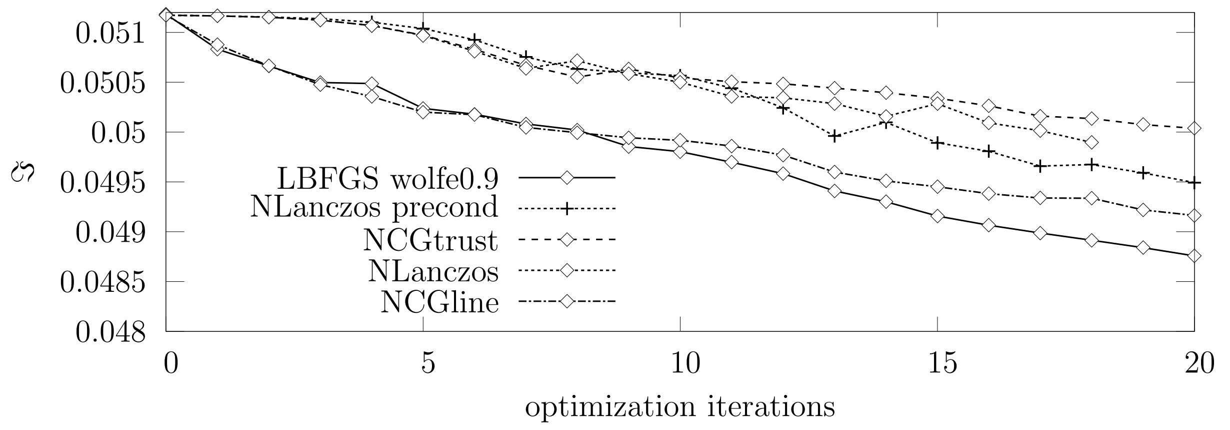
7.1.2. Second Derivative Based Optimization Schemes

The efficiency of optimization algorithms using second derivative information is investigated now. Algorithms NCGline, NCGtrust and NLan, introduced in section 2.3, are used for the noise reduction of case ELES3D. For algorithm NLan a simple preconditioning strategy is tested, too. For the preconditioning the spatially averaged amplitudes of the Hessian vector product, computed during the optimization, were saved at every time. These values give an approximation of the "overall second order sensitivity at some time“ and were used to set the entries of a diagonal preconditioning matrix.

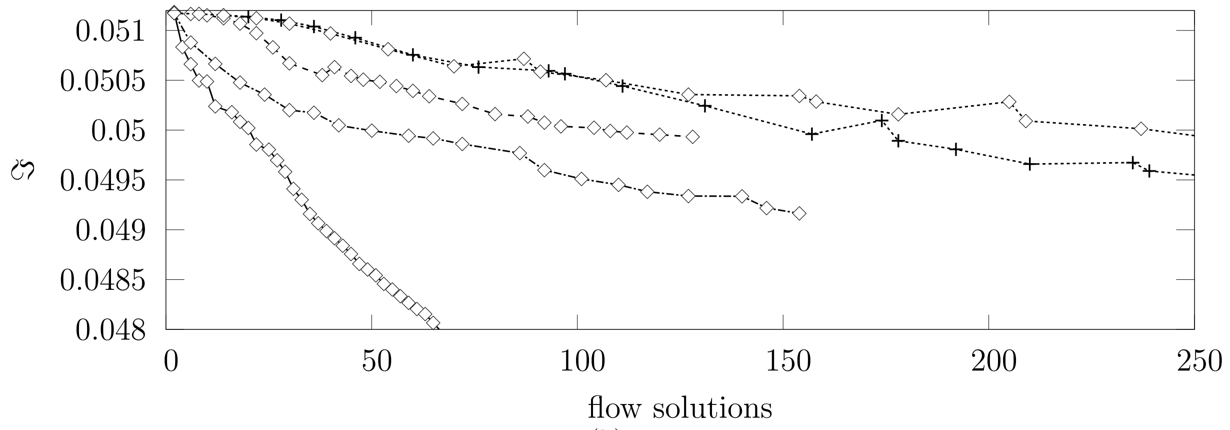
Figure 7.4(a) shows the cost function over the number of optimization iterations for one optimization run. As is to be expected, the NLan, which gives a better approximation to the trust region subproblem (2.14), gives a higher cost functional reduction per iteration compared to the NCGtrust optimization, though the difference is only minimal. The reduction is increased further with the preconditioned NLan scheme. For comparison case LBFGS-Wolfe09 is shown in figure 7.4(a), too. The value for the initial trust region was chosen very small, leading to a small reduction at the beginning of the optimization. Hence, an initial phase with very low reduction can be observed for the trust region methods. After this initial phase, when the trust region is adjusted, the performance per optimization iteration is comparable for the trust region and LBFGS methods. However, a substantial performance advantage of the trust region based optimization schemes can not be observed. This might not be surprising, as e.g. the quadratic convergence properties of Newton like optimization schemes can be proven for situations near a local minimum, only. An assumption surely not fulfilled for the cases in this work.

Scheme NCGline-Wolfe09 performs similar to the LBFGS-Wolfe09 scheme at the beginning of the iterations. This is reasonable as both schemes try to approximate the Newton direction. However, after about 10 iterations the LBFGS based optimization seems to outperform the NCGline scheme. As already mentioned in section 2.3.1, the CGLin iteration performed during the NCGline optimization has to be aborted in case of a search direction with negative curvature. Due to this constraint only about two to three CGLin iterations are performed for one NCGline iteration in most cases. Furthermore, convergence of the CGLin algorithm could not be reached during the performed optimization. Thus, it is questionable if the CGLin scheme gives an accurate approximation to the Newton direction. This might be an explanation why the LBFGS scheme performs better than NCGline.

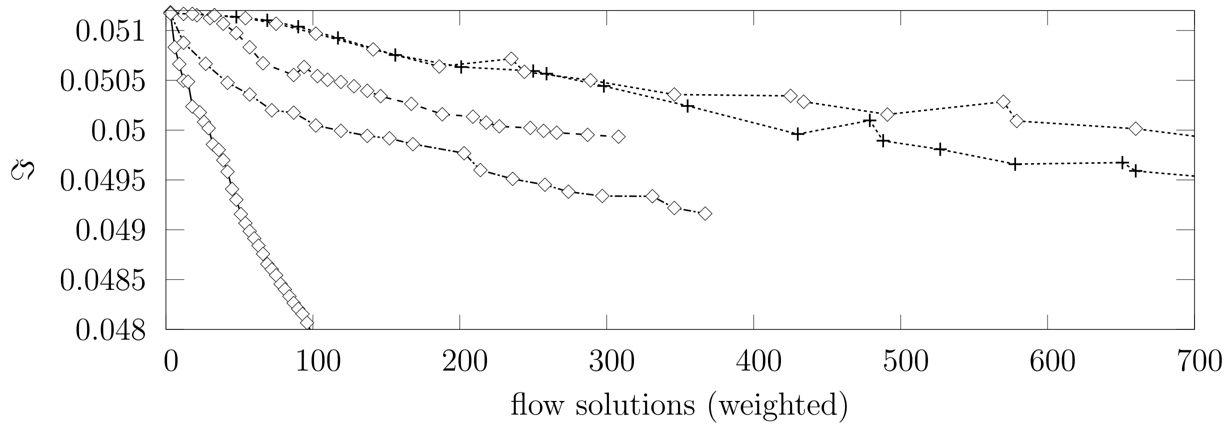
The cost functional is shown over the number of flow solutions in figure 7.4(b) and over a weighted flow count in figure 7.4(c) for the cases described above. In figure 7.4(c) a solution of the sensitivity equations was weighted with a factor of 2.5 and Hessian vector product computation was weighted with 4, to take into account the increased computational cost for the solution of these equations compared to the primal flow equations. As already mentioned in section 2.3.2, the CGLin iteration used during the NCGtrust optimization is



(a)



(b)



(c)

Figure 7.4: Comparison of performance of the Hessian based optimization schemes NCGline, NCGtrust and NLan for case ELES3D.

stopped if a direction of negative curvature is encountered or the search direction leaves the trust region. As a consequence the CGLin iteration was aborted after one or two iterations for the NCGtrust optimization in almost all optimization iterations. On the other hand, the CGLin algorithm used in the NLan optimization was performed till the residual was smaller than 5% of the initial residual, which required between 4 and 12 CGLin iterations. Although it is to be expected that the CGLin algorithm gives only a small improvement to the steepest descent direction after two iterations the lower computational cost of a NCGtrust iteration compared to a NLan iteration is sufficient for the NCGtrust scheme to become more efficient than the NLan scheme, as can be seen in figure 7.4(b) and 7.4(c).

Examining the cost functional reduction in terms of flow solutions the scheme NCGline is clearly outperformed by scheme LBFGS. The reason for this is the expensive CGLin iteration of the NCGline scheme. Furthermore, while for the Wolfe09 line search the first trial step length was excepted in many cases for scheme LBFGS, the step length was initially overpredicted mostly using the NCGline scheme, leading to a higher flow solution count for the determination of a proper step length.

The specific behavior and performance of the optimization runs depends on a number of parameter choices, for example initial flow condition, initial trust region radius, accuracy of line search etc. Furthermore, the behavior of the optimization algorithms could change at later iterations. This might especially hold true if the control reaches the vicinity of a local minimum. This complicates a direct comparison of the different optimization schemes. Nevertheless, the results in figure 7.4 seem clear enough to state that the additional computational effort to determine a search direction, introduced with the Hessian based optimization schemes, is too high to increase the performance of the optimization for the high dimensional non-linear case of this section.

7.2. Optimization Using Exact and Inexact Adjoint

In this section the discrete and continuous adjoint approaches are used for noise optimization to investigate the importance of an accurate gradient direction in an optimization framework.

7.2.1. Two Dimensional DNS

An optimization was performed for case DNS2D for a control horizon of $T = 78$. Comparison with the exact gradient shows a correlation coefficient larger than 0.975 and a deviation according to equation (6.3), which is smaller than 0.1 over the whole interval. Figure 7.5 depicts the cost functional over LBFGS iterations for the continuous and discrete adjoint. It is seen that the reduction is comparable for both cases over the majority of iterations. However, it can be observed that the continuous optimization performs worse around the 10^{th} iteration. This event can be associated with a reset of the LBFGS iteration. The Wolfe line search used in the optimization requires that the proposed search direction is

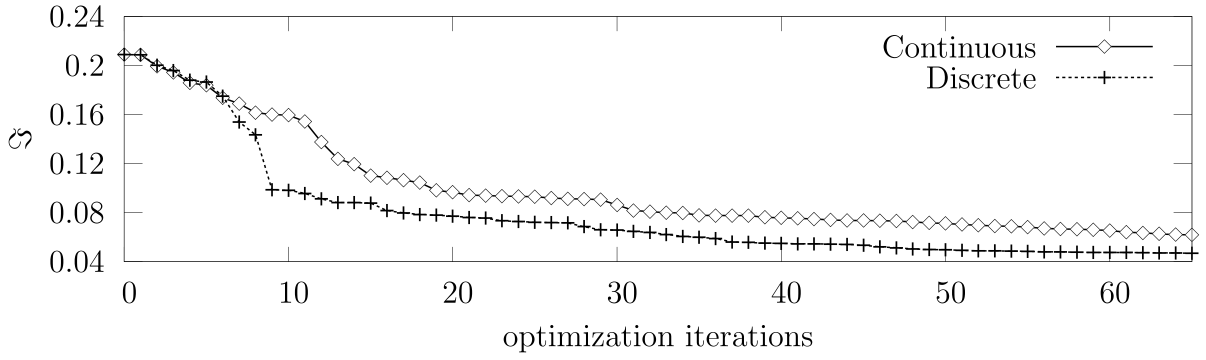


Figure 7.5: Comparison of the cost functional over LBFGS-iterations for case DNS2D involving a short control interval. The optimization is more efficient using the discrete adjoint.

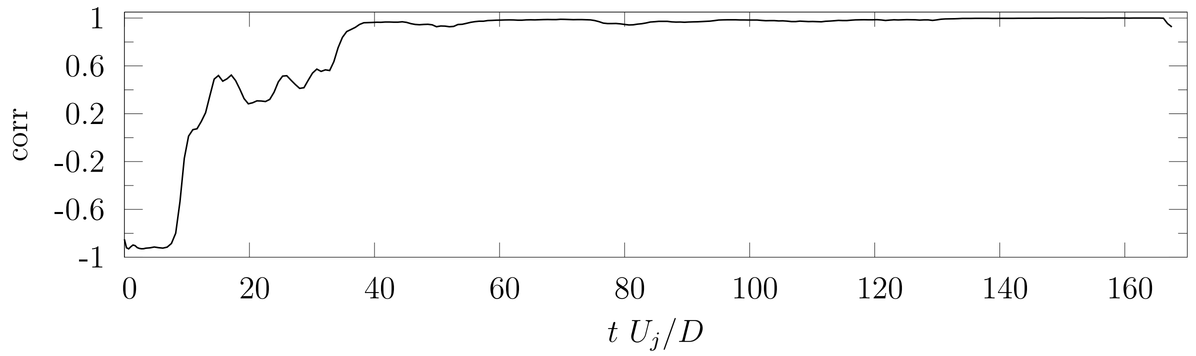


Figure 7.6: Correlation between continuous and discrete adjoint for case DNS2D and a long control interval. The gradient becomes misaligned for times $t \lesssim 40$.

a descent direction, thus the optimization algorithm has to be reset if an ascent search direction occurs. In this work, this case was dealt with by using the simple strategy of discarding information collected during the optimization. Then the procedure is restarted using the control obtained before the occurrence of the ascent direction. Note, that the LBFGS iteration is constructed such that the new search direction is always descent, if the exact gradient is used. Thus, the occurrence of an ascent direction can be linked to inaccuracies in the gradient. Because of this, the reset (and hence the efficiency drop in the optimization) is a consequence of the gradient inaccuracies of the continuous adjoint.

Next, the control horizon is extended up to $\Delta t = 170$. Figure 7.6 shows the correlation according to equation (6.2) between the continuous and discrete adjoint for this long control interval. It can be observed that the continuous adjoint deviates substantially from the exact solution for times $t \lesssim 40$. Therefore, this increased control horizon gives the possibility to investigate the performance of the adjoint optimization in the presence of misaligned gradients.

Using the continuous adjoint, the line search failed to find a step length satisfying the strong

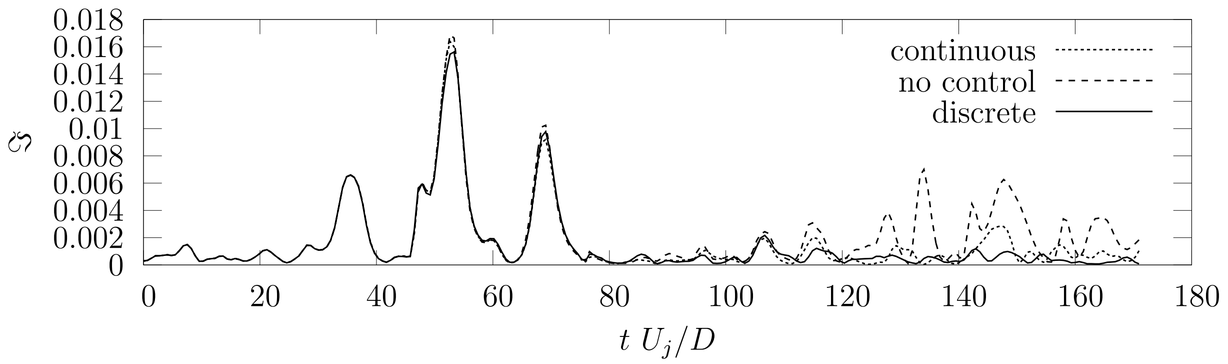


Figure 7.7: Comparison of cost functional over time for the discrete and continuous optimization for case DNS2D involving a long control interval. Both cases fail to significantly reduce the first half of the interval.

Wolfe conditions after the third LBFGS iteration. Since a step length satisfying the strong Wolfe conditions is always obtained as long as the search direction is a descent direction, this is a consequence of the gradient inaccuracies. To be able to perform an optimization using the continuous adjoint a backtracking line search with quadratic interpolation as introduced in section 2.1.1 was used in the following. The backtracking algorithm only requests a sufficient decrease of the cost functional, a condition that could be matched even with an inaccurate gradient.

Figure 7.7 shows the cost function values over time after an optimization run with 60 flow solutions using either the discrete or continuous adjoint. The optimization with the discrete as well as the continuous adjoint shows a significant reduction of the cost functional. However, the discrete adjoint clearly performs better and leads to a higher reduction. One reason for this is that the line search needed to try significantly more step lengths until a sufficient decrease could be obtained using the continuous adjoint. Nevertheless, the reduction achieved by the continuous optimization is remarkable considering the fact that the gradient contains no reliable information over approximately a quarter of the simulation interval. Interestingly, both cases fail to significantly reduce the cost functional for times $t < 80$. This is discussed further in section 7.3.

7.2.2. Three Dimensional LES

Next, an optimization is performed for case ELES3D using the discrete or continuous adjoint approach. For the chosen control horizon of $\Delta t = 70$ (2400 RK-iterations) the correlation coefficient between the discrete and continuous gradient does never drop below 0.995, thus, indicating a very high accuracy of the continuous adjoint. The cost functional is plotted over LBFGS iterations for cases ContF2 and DiscRef in figure 7.8. Surprisingly, the continuous optimization performs better in terms of cost functional reduction, though the difference is only marginal. For further investigation of this observation the discrete

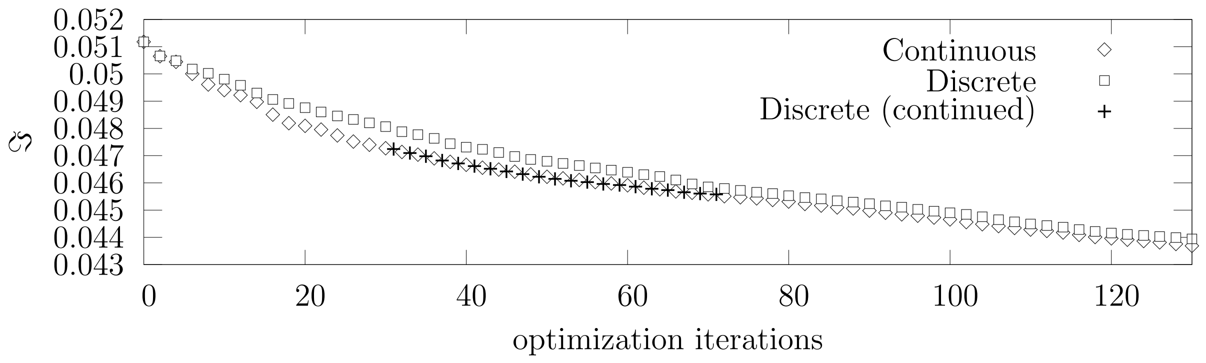


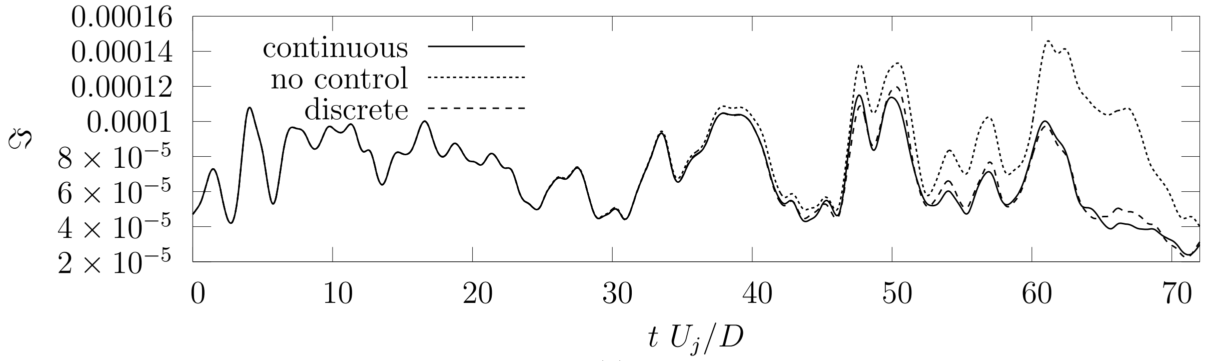
Figure 7.8: Cost-functional over LBFGS-iterations for case ELES3D involving a short control interval. Only every second iteration is shown. The optimization was performed with the continuous and the discrete adjoint approach. Additionally, a discrete optimization was performed starting with a control obtained by the continuous optimization. Both cases successfully reduce the cost functional.

optimization was also based on the solution obtained during the continuous optimization after 30 iterations. This optimization run is shown in figure 7.8 and it becomes apparent that there is only a negligible difference between the continued discrete and the continuous optimization. This behavior supports the conjecture that the differences observed in figure 7.8 are due to the high dimensionality of the control space and thus, the complex appearance of the cost functional surface. This might lead to a different optimization path by chance which doesn't, however, demonstrate a significant advantage of the continuous adjoint approach. Overall it can be summarized that the performance of the optimization is comparable for the continuous and discrete approach in terms of noise reduction and required number of flow solutions. This was to be expected, as the continuous gradient is quite accurate for case ELES3D and short control horizons.

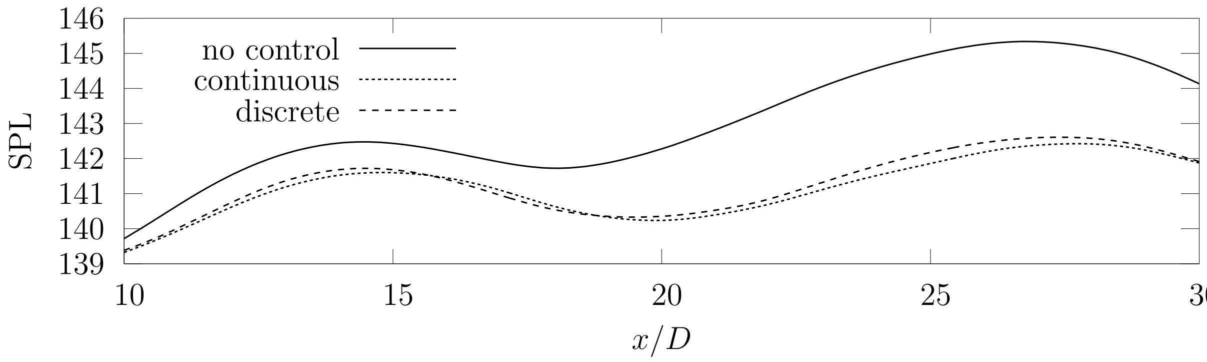
To further illustrate the noise reduction the cost functional is plotted over time for the uncontrolled and controlled cases in figure 7.9(a). Again it becomes clear that the continuous and discrete optimizations perform comparably well. Finally, figure 7.9(b) shows the SPL in decibels plotted over the line Ω defined to be the measure for noise reduction. In this work the SPL is defined as

$$\mathrm{SPL} = 10\ \log_{10}\left(\left(\frac{\overline{p}\ p_{ref}}{p_{min}}\right)^2\right), \tag{7.1}$$

where $\overline{p}$ is the temporally averaged non-dimensional pressure, $p_{min} = 20\mu Pa$ and p_{ref} is given in table 5.1. The reduction ranges from 1 dB to 3 dB depending on the time and position. Not surprisingly, the reduction is highest in the main radiation direction of the jet, where it is loudest.



(a)



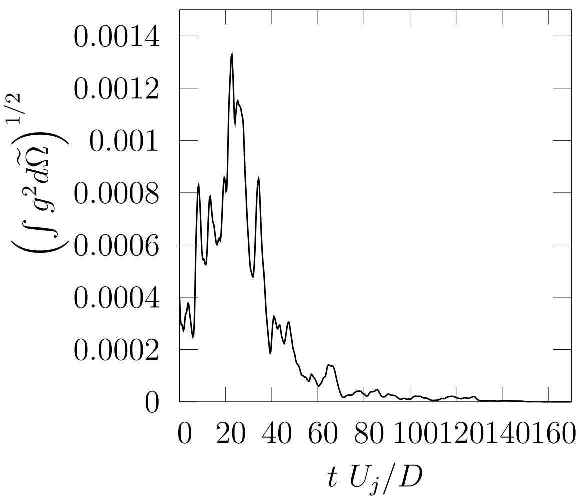
(b)

Figure 7.9: Comparison of (a) cost functional over time and (b) SPL in decibel in streamwise direction for continuous and discrete optimization and uncontrolled case for case ELES3D. A noise reduction about 1 dB to 3 dB can be observed.

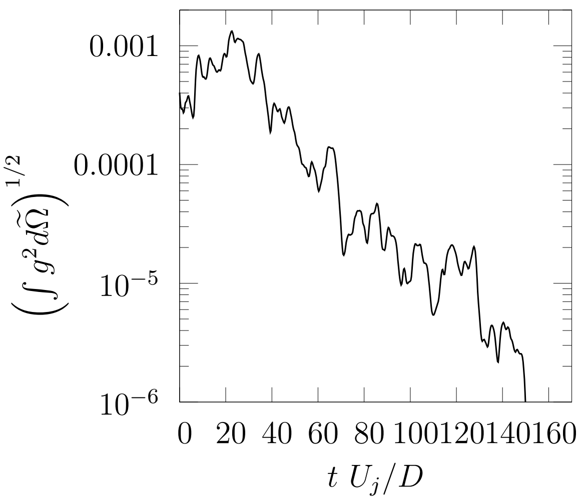
7.3. Long Control Horizons

It was shown in the literature [22, 5] that the choice of the control interval length is of importance for an efficient optimization. This observation is also supported investigating figure 7.7, as an initial interval with $t < 80$ couldn't be successfully controlled even with the knowledge of the exact discrete adjoint using a long control interval of $T = 170$. Note, that this interval could be controlled successfully when optimizing over this shorter interval, only (this was actually done for the case in figure 7.5).

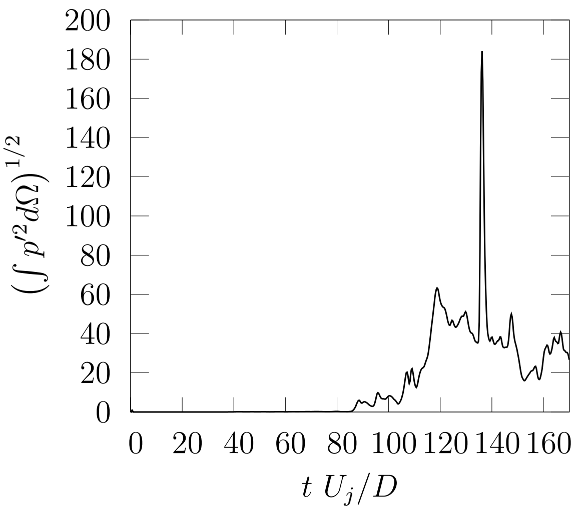
To investigate the problems emerging from long control horizons in more detail figures 7.10(a) and 7.10(b) show the "energy" (in a $L_2 - norm$ sense) of the gradient obtained with control set to zero. It can be observed that the norm of the gradient decreases strongly with time. This strong decrease is a hint that the problem is very ill conditioned and is thus a source for optimization inefficiency. Furthermore, as the control is basically a linear



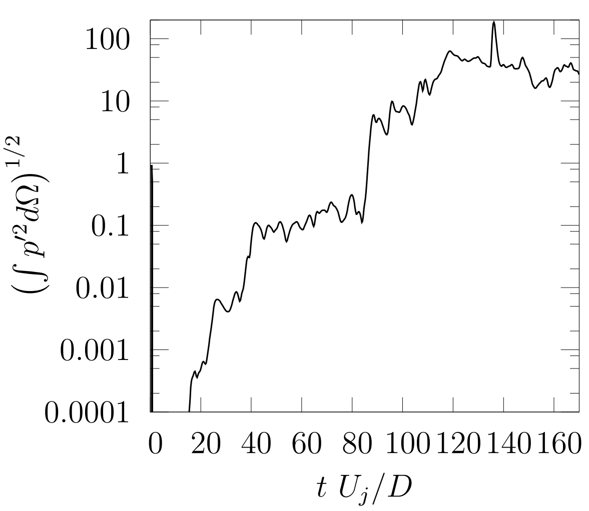
(a)



(b)



(c)



(d)

Figure 7.10: (a) The quantity $\left(\int g^2 \widetilde{\Omega}\right)^{1/2}$ as a measure for the gradient "strength" over time, where g is the gradient and $\widetilde{\Omega}$ is the whole computational domain. (b) Same as (a) but on a logarithmic scale. (c) The quantity $\int p'^2 d\Omega$ as a measure for the linear response of the cost functional to a control perturbation. (d) Same as (c) but on a logarithmic scale. It can be seen that only times $t \gtrsim 100$ are influenced noticeably by the control.

superposition of gradients obtained during the optimization iterations the control has high amplitudes at the beginning of the control interval only and is negligible for the rest of the simulation. Consequently, a significant part of the control interval is practically unused. Another problem with long control horizons becomes apparent by investigating figures 7.10(c) and 7.10(d), which show the quantity $\int p'^2 d\Omega$ as a measure of the strength of the

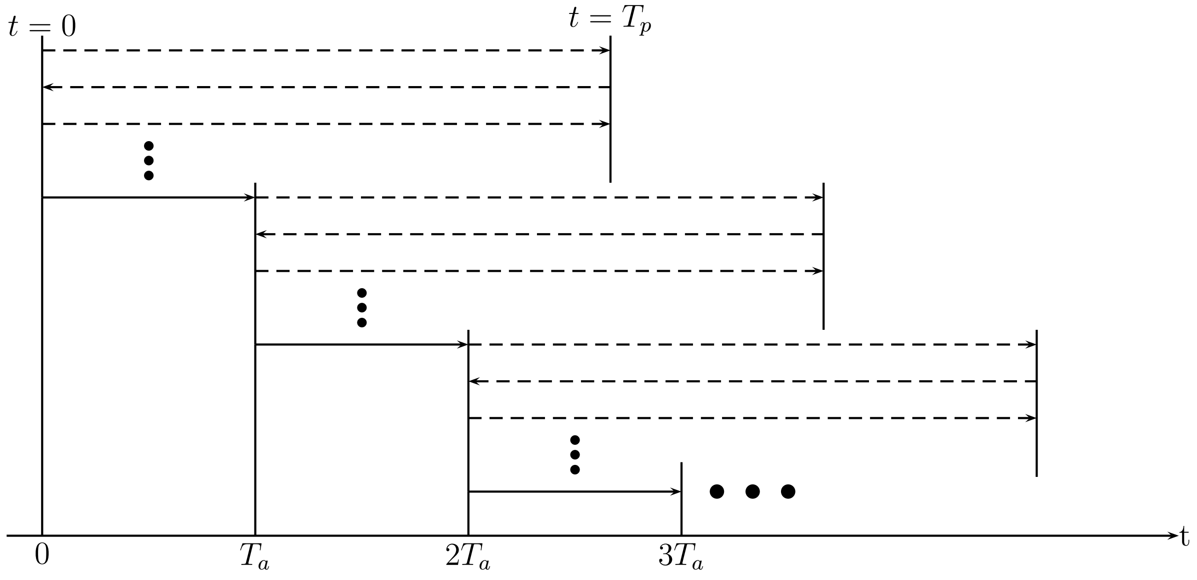


Figure 7.11: Illustration of the receding horizon algorithm: the long interval is split in shorter subintervals and the optimization is carried out on this shorter subintervals [22].

linear response of the cost functional due to a control perturbation. The gradient computed in figure 7.10(a) was chosen as control perturbation ($\mathbf{g}' = \left(\frac{d\Im}{d\mathbf{g}}\right)^T$) and p' constitutes the linear response of the pressure to this perturbation. It can be observed that the cost functionals linear response increases strongly with time. As the gradient used for optimization contains no reliable information for the non-linear regime, the optimization scheme will in most cases select a step size where the linear terms cannot be neglected compared to higher order terms. Thus, the linear response of the cost functional gives an estimate of the change of the cost functional over one optimization iteration. This implies that due to the strong increase of the amplitude of the sensitivities only a short interval at the end of the simulation is actually controlled. It should be noted that the strong amplitude growth is not related to instabilities of the numerics but to instabilities inherent to the linearized Navier-Stokes equations.

The finding that long control intervals are difficult to control even with knowledge of the exact gradient suggests that decomposing the optimization into shorter subintervals might be more efficient. Such an ansatz is realized with the receding horizon algorithm [59, 22] illustrated in figure 7.11. The idea is to minimize the cost functional on a shorter subinterval $T_p < T$. With the control obtained using this minimization the flow is advanced forward some time $T_a \leq T_p$. This procedure is repeated n times until the desired interval length $n\ T_a \geq T$ is reached. A disadvantage of the receding horizon ist that the overall optimization scheme becomes suboptimal, since the optimization is not carried out over the whole control interval. Furthermore, the computational efficiency of the optimization decreases compared to an optimization over the whole control horizon due to the overlapping sub control intervals. On the other hand, this disadvantage might be compensated if the iterative optimization is significantly more efficient on shorter intervals.

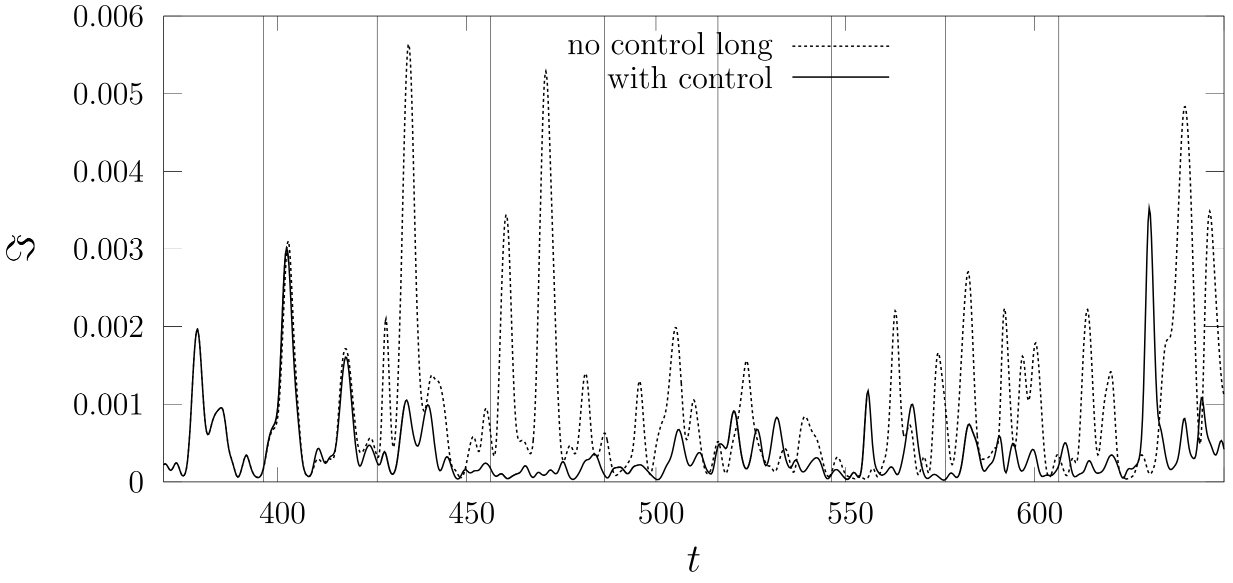


Figure 7.12: Cost functional over time for case DNS2D for the uncontrolled jet and controlled jet obtained from a receding horizon algorithm. Note, that the cost functional doesn't differ at the beginning of the simulation, as the control needs a finite time-interval to effect the sound in the farfield. The vertical bars denote the starting point of the subintervals of the receding horizon algorithm.

Such a receding horizon optimization was performed over 8 subintervals with lengths $T_p = 79$ and $T_a = 30.5$, resulting in an overall control length of $T = 244$. Details of this simulation can also be found in [59]. The cost functional for this case is plotted over time in figure 7.12. Although the less accurate continuous adjoint was used for this optimization the cost function is reduced successfully. It can be seen that, except for a noise peak at $t \approx 625$, the noise peaks are reduced over the whole control interval.

7.4. Influence of Control Space Dimension

Cases DNS2D, ELES3D, LES3D and DNS3D were optimized using the LBFGS-Wolfe09 scheme with control interval length of $T = 70$. The cost function reduction relative to the value without control is compared for these cases in figure 7.13. It becomes obvious that the efficiency of the optimization decreases with increasing resolution for the three dimensional simulations. This is reasonable as the range of scales involved in the flow solution increase with resolution. These smaller scales might reduce the controllability of the numerical simulation. Another explanation for the drop in efficiency with increasing resolution is based on the increased control space dimension.

The volume of the controlled area was kept fixed, such that the different cases are physically comparable. This implies an increasing control space dimension with increasing resolution. Due to the CFL criterion the timestep decreases with increasing resolution and

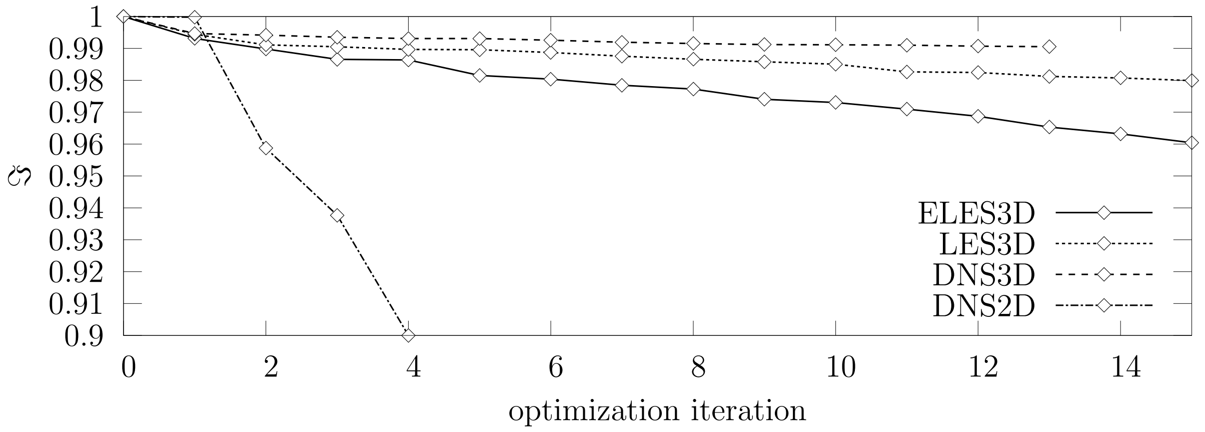


Figure 7.13: Optimization runs with the LBFGS-Wolfe09 scheme for cases DNS2D, ELES3D, LES3D and DNS3D. For better comparison the cost function values are normalized with their initial values. The efficiency of the optimization decreases with increasing resolution.

consequently the number of RK iterations increases. An interpolation point for the control was set every second RK iteration for cases DNS2D, ELES3D and LES3D and every third RK iteration for case DNS3D. Thus, the number of interpolation points in time increases with resolution leading to an increased control space dimension, too. A decrease of the optimization efficiency with decreasing control space dimension also becomes apparent in figure 7.1. For example it can be observed that a simultaneous optimization of stream- and shearwise momentum does perform worse than optimizing only one component of the momentum alone. Apparently, in this case the decreased efficiency due to the control space decrease outweigh the advantage of the increased control possibilites when controlling both momentum components.

A fast cost function reduction of the two dimensional optimization can be observed in figure 7.13 compared to the three dimensional cases. This can be explained with a reasoning similar to the one above comparing the three dimensional cases. First, two dimensional turbulence does not reveal the same complexity as three dimensional turbulence. Secondly, the control space dimension is reduced in the two dimensional simulation, due to the absence of a third spatial direction.

The number of control variables can be reduced by choosing an independent control parameter at only every n^{th} grid point. The missing control function values in between are obtained using interpolation. This kind of control will be referred to as gap$\alpha\beta\gamma$Fδ, where α, β and γ gives the number of grid points between the sampling points for the control in the three spatial directions and δ means that control values are given every δth RK iteration. For example gap111F1 means that sampling control points are given at every grid point in the controlled region and at every RK iteration. The interpolation in the spatial directions is achieved using a Catmull-Rom spline [60]. Controlling only every n^{th} grid point directly means roughly that only wave lengths below an n^{th} of the Nyquist wave

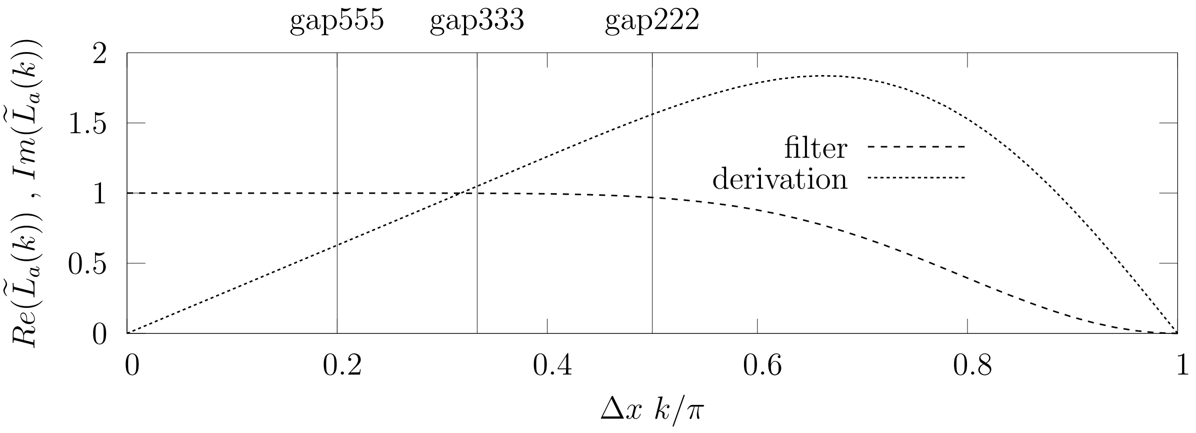
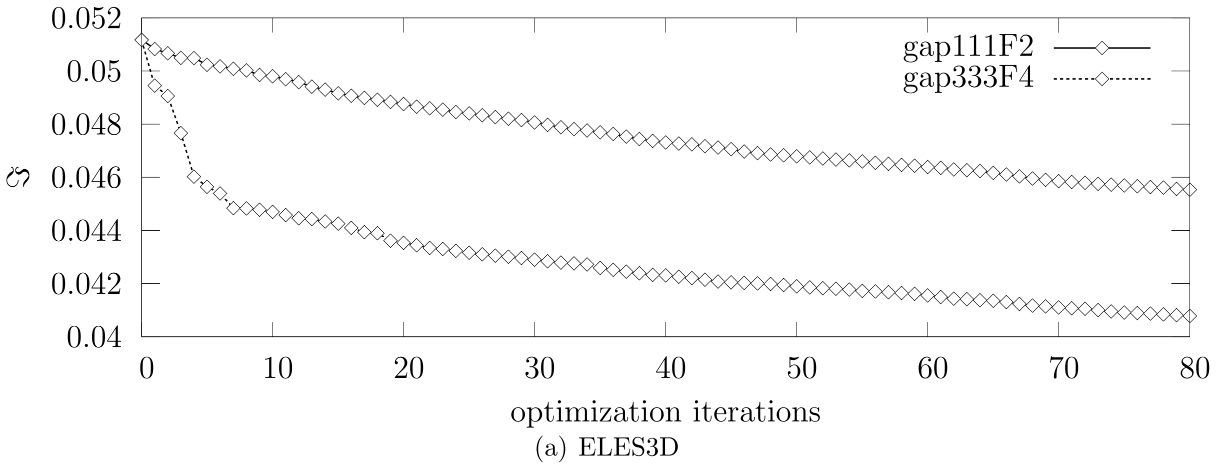


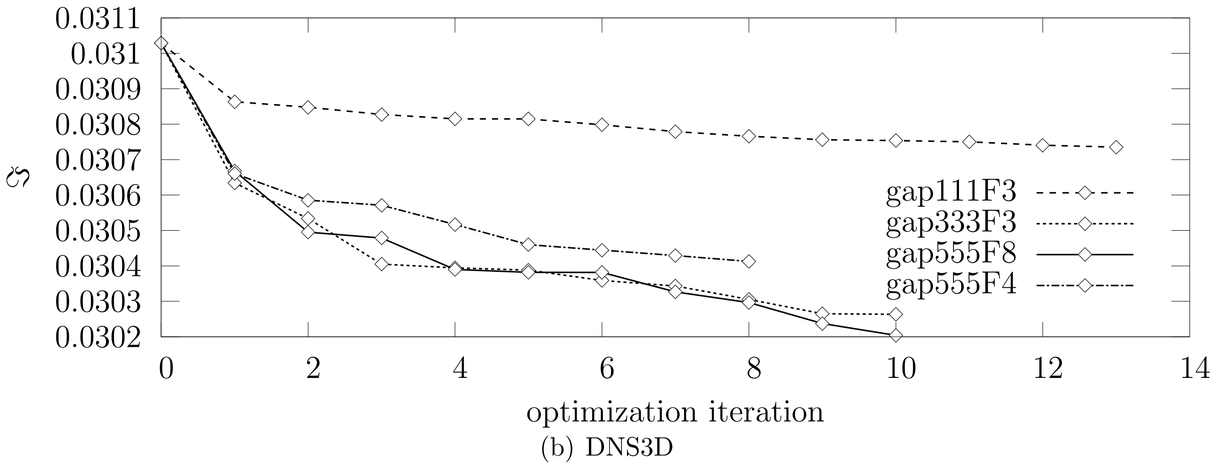
Figure 7.14: Real part of the filter transfer function and imaginary part of the derivative transfer function according to equation (4.11) for the coefficients used in this work. Note, that the other parts of the transfer functions are zero. The horizontal bars give an estimate of the wave numbers which can be influenced directly by a control with a corresponding gap.

lengths can be controlled directly. Consequently, by choosing the gap size between control supporting points there is a trade off between controllability and the reduction in control space dimension. It should be noted, that very high wave numbers are not captured by the FD derivative operators or cut off using filtering. Thus, wave lengths near the Nyquist frequency can not be efficiently controlled anyway, which reduces the frequency band effectively controllable by scheme $gap111$ below the Nyquist frequency. This situation is illustrated in figure 7.14 by plotting the transfer functions of the first derivative and the 10^{th} order filter according to equation (4.11).

Figure 7.15 shows the cost functional over LBFGS iterations for different gaps for cases ELES3D and DNS3D using a control interval length of $T = 70\ D/U_j$. One should have in mind that for case DNS3D due to the computational cost only a moderate number of optimization iterations with only one initial condition were performed. Thus, the specific behavior observed in figure 7.15(b) should be interpreted with care. However, it can be clearly observed that the efficiency of the optimization could be increased successfully by decreasing the control space dimension for both cases. It can be observed in figure 7.15(b) that cases gap333F3 and gap555F8 perform comparably well, whereas the reduction is lower for case gap555F4. This indicates that there isn't a monotonic relation between optimization efficiency increase and control space reduction.



(a) ELES3D



(b) DNS3D

Figure 7.15: Optimizations using the LBFGS-Wolfe09 scheme for cases (a) ELES3D and (b) DNS3D. Different gaps between interpolation points in spatial and temporal directions have been used. Case gap111F2 in (a) and gap111F3 are also shown in figure 7.13. The efficiency of the optimization is increased by a reduction of the number of control variables.

8. Conclusion

A high order finite difference code was developed to solve the compressible unsteady Navier-Stokes equations. The code was used to compute the flow field and the farfield pressure fluctuations of two and three dimensional plane jets with a Reynolds number of 2000 based on inflow velocity and slot width and a Mach number of 0.9. For the three dimensional case three different grids were used in order to investigate the sensitivity of several results with the resolution. Near and farfield statistics were compared to the literature to validate the numerical setup and good agreement could be found, at least for the better resolved cases. In a next step the code was extended to be able to solve the adjoint equations of the Navier-Stokes equations, and its correct implementation could be varified by comparing the response of the adjoint equations due to some control perturbations with results from the sensitivity equations. The discrete and a continuous formulation of the adjoint equations were implemented to be able to directly compare both approaches.

As the discrete adjoint constitutes the exact adjoint system of the discretized system it was used as a reference case to test the validity of several adjoint formulations. The saving frequency of the flow fields, used for the reconstruction of the primal flow variables during the adjoint simulation, was varied to investigate the accuracy of the continuous adjoint in dependence of this parameter. A strong dependence of the gradient accuracy on the saving frequency could be observed. Further investigations have been denoted to the boundary treatment of the continuous adjoint. Accurate results could not be obtained based on sponging as a BC for the adjoint equations alone. The use of appropriate non-reflective adjoint boundary conditions proofed to be necessary, even for cases using a strong sponge at the boundaries. Nevertheless, an increase in accuracy of the continuous adjoint solution could be observed for cases using a strong sponge at the boundaries. However, it appeared to be important to apply the sponge regions consistently in the adjoint and flow solution. To avoid the computation of the transposed linearized Navier-Stokes equations, present in the discrete adjoint formulation, this term was substituted with its continuous counterpart. However, no substantial advantage of this mixed approach could be observed in comparison to the fully continuous adjoint approach. In general the continuous adjoint approach proved to be able to give accurate gradient information. In another test the discrete adjoint formulation was used. However, the recomputation of the flow fields, usually necessitating for adjoint systems of unsteady non-linear systems, was skipped and instead the flow fields were reconstructed using frequent saving and interpolation. Accurate adjoint solutions could be obtained using this approach, provided that the flow field was saved often enough. Especially, using this inexact flow field reconstruction the discrete adjoint formulation was still more accurate than the continuous adjoint. A comparison for the different resolutions revealed that the case with a DNS-like resolution was significantly more accurate than the LES cases.

The two and three dimensional plane jets were used in an optimization framework, aiming for a reduction of the noise emitted by the jet. A three dimensional LES was used to validate the efficiency of different gradient and Hessian based optimization schemes. No convergence to a local minimum could be observed for the optimizations performed in this work. Thus, the cost function reduction served as performance index for the optimization schemes. For conjugate gradient based optimizations an accurate line search proofed to be necessary. For a low storage variant of the BFGS algorithm, on the other hand, it was more efficient to perform a more inaccurate line search, thus increasing the number of optimization iterations per computational time. In both cases a line search based on quadratic/cubic interpolation trying to fulfill the strong Wolfe conditions was more efficient than a bisection-like Brent line search. Although both optimization schemes succeeded in reducing the farfield noise, the LBFGS showed to be slightly more efficient compared to the conjugate gradient scheme. Different optimization schemes which try to approximate the Newton direction with information about a Hessian vector product were used in a line search and trust region based framework. It turned out that Hessian vector product computation performed to improve the search direction is computationally too expensive and the efficiency of the Hessian based optimization were inferior to the gradient based CG and BFGS schemes.

The continuous and discrete adjoint were used for noise optimization in two and three dimensional setups to test the importance of an accurate gradient direction for optimization. The cost functional could be decreased successfully for a two dimensional jet and a short control interval. Nevertheless, the discrete optimization outperformed the continuous optimization due to reasons that could clearly be linked to the gradient inaccuracies. Next, the control interval length was chosen long enough such that the continuous adjoint solution deteriorates substantially from the exact solution near the beginning of the flow simulation. Although a part of the continuous adjoint contained misleading information, the cost functional could successfully be reduced using the continuous adjoint. However, the discrete optimization was clearly superior in terms of reliability and efficiency compared to the continuous approach. For the three dimensional LES of a jet the continuous adjoint showed only minor deviations from the discrete adjoint. Accordingly, both cases could control the jet with comparable efficiency and, although no local minimum could be reached, a noise reduction of up to 3 dB could be achieved.

Increasing the control interval length it could be observed that the optimization failed to reduce the noise at the beginning of the control interval. This problem could be linked to the linear instabilities of the Navier-Stokes equations. A long time interval could be controlled nonetheless using a receding horizon algorithm, which splits the long control intervals into shorter subintervals. It could be observed that the efficiency of the optimization framework decreased with increasing resolution for the chosen volume forcing. The efficiency of the optimization could be increased by introducing distinct supporting points for the control and interpolating in-between, thus reducing the number of control variables.

8.1. Outlook

Although the adjoint equations of the compressible Navier-Stokes equaions could be solved and used for noise optimization for large numerical systems, the research in this field is far from being completed. In this section possible research directions are mentioned, which might result from the findings in this work.

Although the computational cost of determining a gradient is independent of the number of control variables using the adjoint method, the performance of the optimization strongly depends on the control space dimension. The observation that the optimization efficiency could be increased by not controlling every grid point directly indicates that multigrid methods in the control space might be able to increase the efficiency of the optimization in cases where the control space dimension increases with resolution. It was demonstrated that the optimization efficiency decreased with increasing resolution. One important reason for this behavior was the increasing control space dimension of the volume forcing. However, an additional explanation might be that with increasing resolution a broader spectrum of scales participate in the flow solution and have to be controlled, too. More investigations seem necessary to clarify which role smaller scales, not resolved by coarser grids, play in the control and optimization processes. This question might become even more important with increasing Reynolds numbers in combination with LES modeling.

In this work second order adjoint optimization showed to be less efficient than pure gradient based optimization schemes. This might, however, not be true for other flow configurations or other control types. Because of this, it might be worth investigating the efficiency of different optimization schemes for a broader range of flows.

Severe problems were observed for the optimization of long temporal control intervals. These problems also occurred when using the discrete adjoint, which still delivered accurate gradients for long control intervals. Long control horizons could be successfully controlled nonetheless using a receding horizon algorithm. However, this algorithm is strictly speaking a suboptimal approach and involves additional computational cost due to the overlapping subintervals. Because of this alternative approaches to deal with long control horizons should be investigated. One such approach could be a proper preconditioning technique. A preconditioning variant was tested in section 7.1.2 with moderate success, though it should be mentioned that this preconditioning was not tested for long control intervals.

Another possibility might be to manipulate the cost functional, e.g. introducing a time dependent weighting function in the integral of the cost functional. However, further investigation seems necessary to identify suitable preconditioning methods. It could be shown that the problems with long control horizons are connected to the instabilities of the linearized Navier-Stokes equations and the consequential fast grow of the amplitudes of the sensitivity in time. The implications of this finding for time independent control (e.g. shape optimization) with instationary turbulent flows might be worth further investigations. The fast amplitude growth of the adjoint might give the optimizer a hard time to identify a solution optimal for the complete control interval. This might be of special importance as an extension of the receding horizon algorithm to time independent control isn't straight forward.

Another research aim should be to make the optimization results applicable to experiments. One possibility to do so would be to restrict the control parameters to experimentally accessible quantities. For example one might just try to identify an optimal forcing frequency instead of fully time dependent control. Another strategy might be to identify the physical principles of a determined optimal control before trying to mimic the control mechanism instead of applying the numerically optimized control directly. Furthermore, the realism of the computations could be increased by including the nozzle geometry in the simulation.

Bibliography

[1] W Kyle Anderson and V Venkatakrishnan. Aerodynamic design optimization on unstructured grids with a continuous adjoint formulation. *Computers & Fluids*, 28(4):443–480, 1999.

[2] Andreas Babucke, Bruno Spagnoli, Christophe Airiau, Markus Kloker, and Ulrich Rist. Mechanisms and active control of jet-induced noise. In *Numerical Simulation of Turbulent Flows and Noise Generation*, pages 75–98. Springer, 2009.

[3] A. Barbagallo, D. Sipp, and P.J. Schmid. Closed-loop control of an open cavity flow using reduced-order models. *J. Fluid Mech.*, 641:1–50, 2009.

[4] Gal Berkooz, Philip Holmes, and John L Lumley. The proper orthogonal decomposition in the analysis of turbulent flows. *Annual review of fluid mechanics*, 25(1):539–575, 1993.

[5] T. Bewley, P. Moin, and R. Teman. DNS-based predictive control of turbulence: an optimal benchmark for feedback algorithms. *J. Fluid Mech.*, 447:179–225, 2001.

[6] Christian Bischof, Peyvand Khademi, Andrew Mauer, and Alan Carle. Adifor 2.0: Automatic differentiation of fortran 77 programs. *Computational Science & Engineering, IEEE*, 3(3):18–32, 1996.

[7] Daniel Joseph Bodony. *Aeroacoustic Prediction of Turbulent Free Schear Flows.* Dissertation, Stanford University, 2004.

[8] D.J. Bodony. Analysis of sponge zones for computational fluid mechanics. *J. of Comp. Phys.*, 212:681–702, 2005.

[9] Christophe Bogey and Christophe Bailly. Effects of inflow conditions and forcing on subsonic jet flows and noise. *AIAA journal*, 43(5):1000–1007, 2005.

[10] Christophe Bogey and Christophe Bailly. Large eddy simulations of round free jets using explicit filtering with/without dynamic smagorinsky model. *International journal of heat and fluid flow*, 27(4):603–610, 2006.

[11] Christophe Bogey and Christophe Bailly. Large eddy simulations of transitional round jets: Influence of the reynolds number on flow development and energy dissipation. *Physics of Fluids*, 18(6):065101, 2006.

[12] Christophe Bogey and Christophe Bailly. Influence of nozzle-exit boundary-layer conditions on the flow and acoustic fields of initially laminar jets. *Journal of Fluid Mechanics*, 663:507–538, 2010.

[13] Christophe Bogey, Christophe Bailly, and Daniel Juvé. Numerical simulation of sound generated by vortex pairing in a mixing layer. *AIAA journal*, 38(12):2210–2218, 2000.

[14] Christophe Bogey, Olivier Marsden, and Christophe Bailly. Large-eddy simulation of the flow and acoustic fields of a reynolds number 105 subsonic jet with tripped exit boundary layers. *Physics of Fluids*, 23:035104, 2011.

[15] Christophe Bogey, Olivier Marsden, and Christophe Bailly. Influence of initial turbulence level on the flow and sound fields of a subsonic jet at a diameter-based reynolds number of 105. *J. of Fluid Mech.*, 701:352–385, 2012.

[16] J Brezillon and NR Gauger. 2d and 3d aerodynamic shape optimisation using the adjoint approach. *Aerospace Science and Technology*, 8(8):715–727, 2004.

[17] LWB Browne, RA Antonia, S Rajagopalan, and AJ Chambers. Interaction region of a two-dimensional turbulent plane jet in still air. In *Structure of complex turbulent shear flow*, pages 411–419. Springer, 1983.

[18] Mark H. Carpenter, David Gottlieb, and Saul Abarbanel. Time-stable boundary conditions for finite-difference schemes solving hyperbolic systems: Methodology and application to high-order compact schemes. *Journal of Computational Physics*, 111(2):220 – 236, 1994.

[19] G Carpentieri, Barry Koren, and MJL Van Tooren. Adjoint-based aerodynamic shape optimization on unstructured meshes. *Journal of Computational Physics*, 224(1):267–287, 2007.

[20] Gary J Chandler, Matthew P Juniper, Joseph W Nichols, and Peter J Schmid. Adjoint algorithms for the navier–stokes equations in the low mach number limit. *Journal of Computational Physics*, 231(4):1900–1916, 2012.

[21] S Scott Collis and Yong Chang. Computer simulation of active control in complex turbulent flows. *Modeling and Simulation Based Engineering*, 1:851–856, 1998.

[22] S.S. Collis, Y. Chang, S. Kellog, and R.D. Prabhu. Large eddy simulation and turbulence control. *AIAA*, Paper 2000-2564, 2000.

[23] S.S. Collis, K. Ghayour, M. Heinkenschloss, M. Ulbrich, and S. Ulbrich. Optimal control of unsteady compressible viscous flows. *Int. J. for Num. Meth. in Fluids*, 40:1401–1429, 2002.

[24] S.S. Collis, R.S. Joslin, A. Seifert, and V. Theofilis. Active flow control: theory, control, simulation, and experiment. *Progr. in Aerosp. Sci.*, 40:237–289, 2004.

[25] DG Crighton. Basic principles of aerodynamic noise generation. *Progress in Aerospace Sciences*, 16(1):31–96, 1975.

[26] Carlos B da Silva and Olivier Métais. Vortex control of bifurcating jets: A numerical study. *Physics of Fluids*, 14:3798, 2002.

[27] Miguel Fosas de Pando, Denis Sipp, and Peter J. Schmid. Efficient evaluation of the direct and adjoint linearized dynamics from compressible flow solvers. *Journal of Computational Physics*, 231(23):7739 – 7755, 2012.

[28] Karthik Duraisamy and C Praveen. Goal-oriented uncertainty propagation using stochastic adjoints. *Computers & Fluids*, 2012.

[29] R. Ewert and W. Schröder. Acoustic perturbation equations based on flow decomposition via source filtering. *Journal of Computational Physics*, 188:365–398, 2003.

[30] H. Foysi, M. Mellado, and S. Sarkar. Simulation and comparison of variable density round and plane jets. *International Journal of Heat and Fluid Flow*, 31:307–314, 2010.

[31] J.B. Freund. Noise sources in a low-Reynolds-number turbulent jet at Mach 0.9. *J. Fluid Mech.*, 438:277–305, 2001.

[32] Jonathan B Freund, Arnab Samanta, Mingjun Wei, and S Lele. The robustness of acoustic analogies. *AIAA Paper*, 2940, 2005.

[33] Jochen Fröhlich. *Large Eddy Simulation turbulenter Strömungen.* Springer DE, 2006.

[34] Mattia Gazzola, Oleg V Vasilyev, and Petros Koumoutsakos. Shape optimization for drag reduction in linked bodies using evolution strategies. *Computers & Structures*, 89(11):1224–1231, 2011.

[35] Roger G Ghanem and Pol D Spanos. *Stochastic finite elements: a spectral approach.* Courier Dover Publications, 2003.

[36] ME Goldstein. On identifying the true sources of aerodynamic sound. *Journal of Fluid Mechanics*, 526(1):337–347, 2005.

[37] Nicholas Gould, Stefano Lucidi, Massimo Roma, and Philippe Toint. Solving the trust-region subproblem using the lanczos method, 1999.

[38] A. Griewank and A. Walther. revolve: an implementation of checkpointing for the reverse or adjoint mode of computational differentiation. *ACM Transactions on Mathematical Software*, 26, 2000.

[39] Andreas Griewank, David Juedes, and Jean Utke. Algorithm 755: Adol-c: a package for the automatic differentiation of algorithms written in c/c++. *ACM Transactions on Mathematical Software (TOMS)*, 22(2):131–167, 1996.

[40] Max D. Gunzburger. *Perspectives in Flow Control and Optimization.* Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2002.

[41] A. Hay, Borggaard J.T., and Pelletier D. Local improvements to reduced-order models using sensitivity analysis of the proper orthogonal decomposition. *J. Fluid Mech.*, 629:41–72, 2009.

[42] M Heinkenschloss. Numerical solution of implicitly constrained optimization problems. *Rice University Department of Computational and Applied Mathematics Tech. Rep*, 2008.

[43] Michael Hinze and Karl Kunisch. Three control methods for time-dependent fluid flow. *Flow, Turbulence and Combustion*, 65(3-4):273–298, 2000.

[44] M.S. Howe. *Theory of Vortex Sound.* Cambridge University Press, 2003.

[45] F.Q. Hu, M.Y. Hussaini, and J.L. Manthey. Low-dissipation and lowdispersion runge-kutta schemes for computational acoustics. *J. Comput. Phys.*, 124:177–197, 1995.

[46] K. Ito and K. Kunisch. *Lagrangian Multiplier Approach to Variational Problems and Applications.* Society for Industrial and Applied Mathematics, Philadelphia, PA, USA, 2008.

[47] Armin Ghajar Jazi. First- and second-order adjoint based optimal control of a plane jet. Master's thesis, RWTH Aachen University, Germany, 2012.

[48] Stefan Johansson. High order finite difference operators with the summation by parts property based on DRP schemes. Technical Report 2004-036, it, August 2004.

[49] F. Kerherv, P. Jordan, A. V. G. Cavalieri, J. Delville, C. Bogey, and D. Juv. Educing the source mechanism associated with downstream radiation in subsonic jets. *Journal of Fluid Mechanics*, 710:606–640, 10 2012.

[50] J. Kim, D.J. Bodony, and J.B. Freund. A High-Order, Overset-Mesh Algorithm for Adjoint-Based Optimization for Aeroacoustics Control. *AIAA J.*, AIAA 2010-3818, 2010.

[51] Randall R Kleinman and Jonathan B Freund. The sound from mixing layers simulated with different ranges of turbulence scales. *Physics of Fluids*, 20:101503, 2008.

[52] Omar M Knio, Habib N Najm, Roger G Ghanem, et al. A stochastic projection method for fluid flow: I. basic formulation. *Journal of Computational Physics*, 173(2):481–511, 2001.

[53] Dana A Knoll and David E Keyes. Jacobian-free newton–krylov methods: a survey of approaches and applications. *Journal of Computational Physics*, 193(2):357–397, 2004.

[54] G. Kuruvila, S. Ta, and M. D. Salas. Airfoil optimization by the one-shot method. *NASA Langley Technical Report Server, Agard-94-803*, 1994.

[55] OP Le Maître and L Mathelin. Equation-free model reduction for complex dynamical systems. *International Journal for Numerical Methods in Fluids*, 63(2):163–184, 2010.

[56] S. K. Lele. Compact Finite Differences Schemes with Spectral-like Resolution. *J. Comput. Phys.*, 103:16–42, 1992.

[57] Guido Lodato, Pascale Domingo, and Luc Vervisch. Three-dimensional boundary conditions for direct and large-eddy simulation of compressible viscous flows. *Journal of Computational Physics*, 227(10):5105 – 5143, 2008.

[58] Ali Mani. Analysis and optimization of numerical sponge layers as a nonreflective boundary treatment. *Journal of Computational Physics*, 231(2):704 – 716, 2012.

[59] Daniel Marinc and Holger Foysi. Investigation of a continuous adjoint-based optimization procedure for aeroacoustic control of plane jets. *International Journal of Heat and Fluid Flow*, 38(0):200 – 212, 2012.

[60] Stephen R Marschner and Richard J Lobb. An evaluation of reconstruction filters for volume rendering. In *Proceedings of the conference on Visualization'94*, pages 100–107. IEEE Computer Society Press, 1994.

[61] A. L. Marsden, M. Wang, J. E. Jr. Dennis, and P. Moin. Trailing-edge noise reduction using derivative-free optimization and large-eddy simulation. *J. Fluid Mech.*, 572:13–36, 2007.

[62] M.P. Martin, U. Piomelli, and G.V. Candler. Subgrid-scale models for compressible large-eddy simulations. *Theoret. Comp. Fluid Dyn.*, 13:361–376, 2000.

[63] L Mathelin and OP Le Maıtre. Robust control of uncertain cylinder wake flows based on robust reduced order models. *Computers & Fluids*, 38(6):1168–1182, 2009.

[64] Lionel Mathelin, Luc Pastur, and Olivier Le Maître. A compressed-sensing approach for closed-loop optimal control of nonlinear systems. *Theoretical and Computational Fluid Dynamics*, 26(1-4):319–337, 2012.

[65] J. Mathew, R. Lechner, H. Foysi, J. Sesterhenn, and R. Friedrich. An explicit filtering method for large eddy simulation of compressible flows. *Phys. of Fluids*, 15(8):2279–2289, 2003.

[66] J.J. More and D.J. Thuente. Line Search Algorithms with Guaranteed Sufficient Decrease. *ACM Transactions on Mathematical Software*, 20(3):286–307, 1994.

[67] Habib N Najm. Uncertainty quantification and polynomial chaos techniques in computational fluid dynamics. *Annual Review of Fluid Mechanics*, 41:35–52, 2009.

[68] Eric J. Nielsen, Boris Diskin, and Nail K. Yamaleev. Discrete adjoint-based design optimization of unsteady turbulent flows on dynamic unstructured grids. *AIAA Journal*, 48(6):1195–1206, 2010.

[69] Eric J. Nielsen and William L. Kleb. Efficient construction of discrete adjoint operators on unstructured grids using complex variables. *AIAA Journal*, 44(4):827–836, 2006.

[70] J. Nocedal and S. Wright. *Numerical Optimization*. Springer, 2006.

[71] Jacques E.V. Peter and Richard P. Dwight. Numerical sensitivity analysis for aerodynamic optimization: A survey of approaches. *Computers & Fluids*, 39(3):373 – 391, 2010.

[72] T.J. Poinsot and S.K. Lele. Characteristic boundary conditions. *J. Comput. Phys.*, 101:104, 1992.

[73] S. B. Pope. *Turbulent Flows*. Cambridge University Press, 2000.

[74] Press, W.H. and Flannery, B.P. and Teukolsky, S.A. and Vetterling, W.T. *Numerical Recipes*. Cambridge University Press, 1986.

[75] Jan Riehme, Andrea Walther, Jörg Stiller, and Uwe Naumann. *Adjoints for time-dependent optimal control*. Springer, 2008.

[76] Clarence W Rowley, Tim Colonius, and Richard M Murray. Model reduction for compressible flows using pod and galerkin projection. *Physica D: Nonlinear Phenomena*, 189(1):115–129, 2004.

[77] Markus P Rumpfkeil and Dimitri J Mavriplis. Efficient hessian calculations using automatic differentiation and the adjoint method with applications. *AIAA journal*, 48(10):2406–2417, 2010.

[78] Markus P Rumpfkeil and David W Zingg. A hybrid algorithm for far-field noise minimization. *Computers & Fluids*, 39(9):1516–1528, 2010.

[79] M. Samimy, J.-H. Kim, J. Kastner, I. Adamovich, and Y. Utkin. Active control of high-speed and high-reynolds-number jets using plasma actuators. *Journal of Fluid Mechanics*, 578:305–330, 4 2007.

[80] Neil D Sandham, CL Morfey, and ZW Hu. Nonlinear mechanisms of sound generation in a perturbed parallel jet flow. *Journal of Fluid Mechanics*, 565(1), 2006.

[81] Peter J Schmid. Dynamic mode decomposition of numerical and experimental data. *Journal of Fluid Mechanics*, 656(1):5–28, 2010.

[82] Jan Schulze. *Adjoint based jet-noise minimization.* Dissertation, Technische Universität Berlin, 2012.

[83] Samuel Sinayoko, Anurag Agarwal, and Z Hu. Flow decomposition and aerodynamic sound generation. *Journal of Fluid Mechanics*, 668:335–350, 2011.

[84] B Spagnoli and Christophe Airiau. Adjoint analysis for noise control in a two-dimensional compressible mixing layer. *Computers & Fluids*, 37(4):475–486, 2008.

[85] DN Srinath and Sanjay Mittal. An adjoint method for shape optimization in unsteady viscous flows. *Journal of Computational Physics*, 229(6):1994–2008, 2010.

[86] D Stanescu and WG Habashi. 2 n-storage low dissipation and dispersion runge-kutta schemes for computational acoustics. *Journal of Computational Physics*, 143(2):674–681, 1998.

[87] S.A. Stanley, S. Sarkar, and J.P. Mellado. A study of the flowfield evolution and mixing in a planar turbulent jet using direct numerical simulation. *J. Fluid Mech.*, 450:377–407, 2002.

[88] Scott Stanley and Sutanu Sarkar. Simulations of Spatially Developing Two-Dimensional Shear Layers and Jets. *Theoretical and Comp. Fluid Dynamics*, 9:121–147, 1997.

[89] S. Stolz, N. A. Adams, and L. Kleiser. An approximate deconvolution model applied for large-eddy simulation with application to incompressible wall-bounded flows. *Phys. Fluids*, 13:997–1015, 2001.

[90] S. Stolz and N.A. Adams. An approximate deconvolution procedure for large-eddy simulation. *Phys. Fluids*, 11:1699–1701, 1999.

[91] B. Strand. Summation by parts for finite difference approximations for d/dx. *J. Comput. Phys.*, 110:47–67, 1994.

[92] Victoria Suponitsky, Neil D Sandham, and Christopher L Morfey. Linear and nonlinear mechanisms of sound radiation by instability waves in subsonic jets. *Journal of Fluid Mechanics*, 658:509–538, 2010.

[93] Eka Suwartadi, Stein Krogstad, and Bjarne Foss. Second-order adjoint-based control for multiphase flow in subsurface oil reservoirs. In *Decision and Control (CDC), 2010 49th IEEE Conference on*, pages 1866–1871. IEEE, 2010.

[94] C. K. W. Tam and J. C. Webb. Dispersion-relation-preserving finite difference schemes for computational acoustics. *J. Comput. Phys.*, 107:262–281, 1993.

[95] Christopher KW Tam, K Viswanathan, KK Ahuja, and J Panda. The sources of jet noise: experimental evidence. *Journal of Fluid Mechanics*, 615(1):253–292, 2008.

[96] FO Thomas and HC Chu. An experimental investigation of the transition of a planar jet: Subharmonic suppression and upstream feedback. *Physics of Fluids A: Fluid Dynamics*, 1:1566, 1989.

[97] FO Thomas and KMK Prakash. An experimental investigation of the natural transition of an untuned planar jet. *Physics of Fluids A: Fluid Dynamics*, 3:90, 1991.

[98] K.W. Thompson. Time Dependent Boundary Conditions for Hyperbolic Systems. *J. of Comp. Phys.*, 68:1–24, 1986.

[99] Miguel R Visbal and Datta V Gaitonde. On the use of higher-order finite-difference schemes on curvilinear and deforming meshes. *Journal of Computational Physics*, 181(1):155–185, 2002.

[100] B. Vreman, B. Guerts, and H. Kuerten. Large-eddy simulation of the turbulent mixing layer. *J. Fluid Mech.*, 339:357–390, 1997.

[101] Andrea Walther, Andreas Griewank, and Olaf Vogel. Adol-c: Automatic differentiation using operator overloading in c++. *PAMM*, 2(1):41–44, 2003.

[102] Zhi Wang, K Droegemeier, and L White. The adjoint newton algorithm for large-scale unconstrained optimization in meteorology applications. *Computational Optimization and Applications*, 10(3):283–320, 1998.

[103] M. Wei and J. B. Freund. A noise-controlled free shear flow. *Journal of Fluid Mechanics*, 546:123–152, 2006.

[104] Mingjun Wei. *Jet Noise Control by Adjoint-Based Optimization.* Dissertation, University of Illinois, 2004.

[105] David C. Wilcox. *Turbulence Modeling for CFD.* DCW Industries, Inc., La Canada, California, 1994.

[106] C.C.L. Yuan, M. Krstic, and Bewley T.R. Active control of jet mixing. *IEE Proceedings*, 151:763–772, 2004.

[107] A.S. Zymaris, D.I. Papadimitriou, K.C. Giannakoglou, and C. Othmer. Adjoint wall functions: A new concept for use in aerodynamic shape optimization. *Journal of Computational Physics*, 229(13):5228 – 5245, 2010.