Seventh International Conference on Computational Fluid Dynamics (ICCFD7), Big Island, Hawaii, July 9-13, 2012

ICCFD7-1901

Use of Evolutionary Algorithm in the Analysis and Optimisation of Turbulence Models

B. Fabritius* and G. Tabor*
Corresponding author: b.fabritius@exeter.ac.uk

* University of Exeter, UK

Abstract: In this paper we use a genetic algorithm to identify an optimised set of model parameters for a turbulence model to improve its performance in CFD simulations. We look at two test cases where good experimental data was available and use this data to train the model coefficients towards the flow problem at hand. We can show a significant performance boost using the optimised parameters with respect to the velocity profiles of the flow. The focus lies primarily on the k-ε and the k-ω SST models for the first case and the Spalart-Allmaras one-equation model for the latter. The influence of each arbitrary parameter on the development of the flow is investigated. After identification of the most influential parameters an optimisation is performed. The best set of coefficients is determined per test case. The optimisation method is based on evolutionary computation principles using an elitist genetic algorithm. The optimised set of coefficients can then be used to solve flow problems of similar configuration to a higher accuracy than by using the standard values.

Keywords: Numerical Algorithms, Computational Fluid Dynamics, Turbulence Modeling.

1 Introduction

Numerical simulation of complex flow phenomena is a challenging field in fluid dynamics. Even with computers getting faster and massively parallel in recent years, the accuracy of the computations is still dependent on the models that describe the underlying flows. The execution of a direct numerical simulation (DNS) is still far away from being affordable in terms of computation time. Grid spacing and time discretization scale with power laws of the Reynolds number and realistic cases are far beyond the capabilities even of modern supercomputers [1]. While in the 80s and early 90s of the last century memory was the limiting factor, it is now more likely to be time. For example Erturk et. al. [2] reported the largest computable Reynolds number for the relatively simple 2D lid-driven cavity flow on a 600×600 grid to be 21,000. That is why most solvers seek to solve the Reynolds-Averaged Navier-Stokes (RANS) equations described in section 4.1, where the main flow velocity is separated into a mean velocity component and turbulent fluctuations, expanding the NS-equations by additional terms that need to be modelled. Several different approaches have been developed and applied, ranging from one-equation models like the Spalart-Allmaras model [3], and two-equation models like the k-ε model by [4] or the k-ω model by [5], as well as various combinations and variations of these. The empirical nature of these formulations can lead to undesired behaviour, especially in very heterogenous flows or flow regions with highly unsteady turbulent fluctuations. To improve the applicability of all the models to as wide a range of problems as possible several improvements and changes have been proposed. A simple internet search reveals hundreds of different models, some only slight modifications to the most common ones, others adjusted to specific flow types like, for example, flow around buildings, oceanic flow, flow through porous media and many others. Common to all of these models is a formulation that contains a set of arbitrary parameters that need to be determined either by experimental observation or by describing relations under simplified conditions. The way in which the values are obtained is up to the developer and closely related to the choice of flows that are used for their determination. The ideal case would be to find an argument for each closure coefficient separately, but in reality two or more are involved and are rarely independent.

2 Evolutionary Computation

2.1 Overview

Evolutionary Computation (EC) can be divided into three major categories: Genetic Algorithms (GA) developed by Holland [6] with contributions by deJong, Goldberg and others, Evolution Strategies first mentioned in Germany by Rechenberg and Schwefel [7, 8] and Evolutionary Programming described initially by Fogel, Owens and Walsh [9] in the United States. All three have since then spawned a series of journals and conferences related to the topic and have been further developed in recent years. Optimization procedures, as used in this work, belong to the category of Genetic Algorithms. The general principle of all flavours of EC is that a solution to a problem 'evolves' from a randomly generated initial population in a series of generations. By application of the three evolutionary operators: selection, crossover and mutation, changes are made to the individuals that ultimately lead to a better fitness. This fitness is a measure of the quality of a solution. Good results have been demonstrated in difficult optimisation problems that are either impenetrable by standard methods, such as NP-complete problems, or that are expensive in terms of evaluation [10]. The behaviour of GAs can be influenced in different ways. There are, for example, different selection methods to choose from or the probabilities that control the evolutionary process can be modified.

Many large-scale optimisation problems can only be solved approximately. Genetic Algorithms are a good choice for these kind of problems. They combine stochastic with direct search, making them very robust to the topology of the solution space. GAs work on a broad data basis at all times, while pure deterministic optimisers tend to concentrate on a single solution. Early studies of the performance of evolutionary optimisation for a set of a canonical topologies were carried out by deJong [11]. In an engineering context GAs have been applied to a wide range of applications. Any optimisation problems that require expensive computations are possible candidates for evolutionary methods. Most of the work that has been published up until now has been design related. In electrical engineering, for example, to design electric circuit boards [12] or in hydromechanics in the planning of large-scale water distribution systems [13]. In conjunction with CFD the evolutionary approach has mainly been used for shape optimisation, e.g. in the design of motor fan blades [14] or heat-exchanger blades [15]. To our knowledge the use of these techniques to improve the accuracy of CFD modelling represents a novel approach.

2.2 Methodology

Genetic Algorithms are based on the principle of natural selection and natural genetics [10]. GAs are randomly initialised, asserting a diverse set of possible solutions. Compared to conventional optimisation methods they will climb many peaks simultaneously during the evolution process. That reduces the probability to concentrate on the wrong peak representing a local optimum, as common gradient based methods would do. Figure 1 depicts the sequence of operations in a typical GA. A set of parameters in a GA will generally be coded as a string of finite length, most commonly a binary string. Each of these strings (also *chromosome* or *genotype*) represents one possible solution to the optimisation problem. Two opposed strategies are at work here: Exploitation of a single solution versus exploration of the solution space. Classical gradient based methods concentrate on the former, while sole usage of the latter would correspond to a random search. GAs manage to reach a surprisingly good balance between those two extremes [16]. Each population undergoes a simulated evolution. Good solutions reproduce while less favourable solutions are discarded (or 'die'). Individuals are selected for reproduction depending on their fitness value. This selection process is stochastically controlled, assigning fitter individuals a higher probability to get chosen. From those individuals (*parents*) selected in this manner, offspring (*children*) are generated by applying crossover and mutation operators. The crossover operator uses two parents and combines elements from one parent with elements from the other, creating a new individual that now contains information from both its ancestors.

An example of single point crossover between two chromosomes (binary strings) a and b of length n+1:

$$\begin{aligned} a &= \langle a_n\, a_{n-1}\, \ldots\, a_1\, a_0 \rangle \\ b &= \langle b_n\, b_{n-1}\, \ldots\, b_1\, b_0 \rangle \end{aligned}$$

with a randomly selected crossover point $X \epsilon [0, n-2]$, creating children:

$$\begin{aligned} a' &= \langle a_n\, a_{n-1}\, \ldots\, a_{X+1}\, b_X\, b_{X-1} \ldots\, b_1\, b_0 \rangle \\ b' &= \langle b_n\, b_{n-1}\, \ldots\, b_{X+1}\, a_X\, a_{X-1} \ldots\, a_1\, a_0 \rangle \end{aligned}$$

Mutation is in most cases implemented as bitwise mutation where the value of a single bit in a chromosome is inverted. The probability of mutation or crossover occuring is controlled by external variables P_M and P_C respectively. Other parameters that influence the performance of the GA are the population size S and the number of generations G. In the optimisation problem at hand, the multiple real values are bit-string encoded and the fitness objectives are measurable properties of the flow.

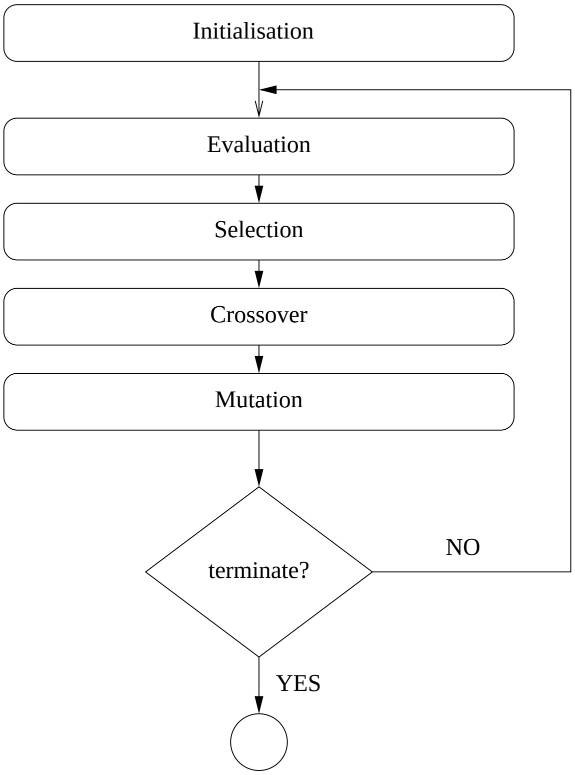


Figure 1: Schematic of the workflow of a typical GA

2.3 Parameter coding

Since the problem variables are real values and their chromosomal representation is a binary string, a mapping has to be defined. For a single coefficient $c \,\epsilon\, [c_{lo}, c_{hi}]$ the length of the bitfield has to be determined by taking into account the desired resolution Δ_c of the interval. The number of bits required is now

$$n = \left\lceil \log_2 \left(\frac{c_{hi} - c_{lo}}{\Delta_c} + 1 \right) - 1 \right\rceil \tag{1}$$

Translation from binary to decimal values can now easily be done as follows:

$$\langle b_n\, b_{n-1}\, \ldots\, b_1\, b_0 \rangle_2 \; = \; \left(\sum_{i=0}^{n} b_i \cdot 2^i \right)_{10} = c' \tag{2}$$

$$c \; = \; c_{lo} + c' \cdot \frac{c_{hi} - c_{lo}}{2^{n+1} - 1} \tag{3}$$

2.4 Multi-objective Optimisation

Complex optimisation problems often seek to find optimal solutions with respect to multiple, often concurring, objectives. Many multi objective evolutionary algorithms (MOEAs) have been developed in the last few decades [17, 18, 19]. Since it is generally the case that a problem has no single solution that is optimal w.r.t. all objectives simultaneously, a number of equally optimal solutions are created that lie on the Pareto-optimal front. The algorithm that is used in this study is a fast elitist non-dominated sorting genetic algorithm (NSGA), that was originally introduced by Srinivas and Deb [17] and improved by Deb et. al. [20]. The second generation version NSGA-II removed some of the criticised flaws in the original algorithm and is able to capture high order Pareto surfaces.
Elitism speeds up the convergence of the GA and prevents the loss of the best solutions. The sorting procedure orders solutions by the level of dominance over concurring solutions. That way the most dominant individuals are considered to be the fitter ones and therefore have a higher chance to contribute to the next generation. The algorithm was successfully used in engineering optimisation problems [21, 22]. The implementation of NSGA-II and the application to the optimisation of turbulence model coefficients is currently under development and results will be published subsequently.

3 Implementation

3.1 Software Packages

Available for OpenFOAM is a toolset called pyFoam [1] written in the object-oriented language Python. It offers applications to read, modify and run OpenFOAM cases as well as analyse the results. Inspired by this, the framework for the evolutionary computation capabilities is developed in Python. That way the envoking and manipulation functions provided by pyFoam can be used and execution of the program can easily be controlled by using scripts. One of the most important requirements in the development of an EC software in the context of CFD is the capability to parallelise the code to allow for faster computation spread over several processing units. A commonly used library to realise this is the MPI (Message Passing Interface) standard [23]. The Python implementation named mpi4py is used in the current project. While it is not a full realisation of the MPI standard, it provides all the required functions for the purpose of this research.

3.2 Code Design

In order to write software that is as generic as possible the design process has to be treated with special care. Based on the guidelines by Gagné and Parizeau on how to write generic EC software tools [24], the framework structure should meet these minimal criteria:
See the reference for details about how to measure the fulfilment of these goals. The term 'generic' in this context needs further explanation. According to the computer dictionary[2], generic software is *'Software which can perform many different types of tasks but is not specifically designed for one type of application.'* Taking that into account the development of a generic EC framework should not be tailored to one specific form of optimisation. Operators, such as the crossover or selection operator, should be interchangeable regardless of the objects they are applied to. In addition the underlying representation of a solution should not affect the way the GA works. Interchangeability of operators can easily be implemented in modern object-oriented programming languages. The user can choose at run-time between a given set of predefined

[1] http://openfoamwiki.net/index.php/Contrib_PyFoam
[2] http://www.computingstudents.com/dictionary

operators or can add new operators to meet specific needs. This is usually the case for the fitness evaluation which is a problem dependent function. Reusability and independence of the optimisation problem on top are key features of the selection and crossover mechanisms. Commonly used realisations of these are therefore included in the developed framework, but can be altered or new ones implemented. This is possible through the realisation of the *strategy* design pattern (see chapter 5 in [25]). Equally flexible is the selection of the coding algorithm that encodes and decodes the chromosome as described in section 2.3.
In the developed software package control parameters can be set using external configuration files. For every variable that is subject to the evolution process the user can define lower and upper bounds as well as the desired precision. This allows running different test cases with different initial setups without altering the code. The only element that has to be adapted for each case is the fitness evaluation function since it is problem dependent.

4 Model Equations

4.1 RANS modelling

If a flow is statistically steady it is possible to decompose the flow variables ϕ into an ensemble averaged part $\overline{\phi}$ and fluctuations ϕ' about that average. This process is known as *Reynolds Averaging* [26] and when applied to the Navier-Stokes equations leads to Reynolds-Averaged Navier-Stokes (RANS) equations. In case of incompressible flow with body forces, the continuity and momentum equations can be written as

$$\frac{\partial(\rho\overline{u}_i)}{\partial x_i} = 0, \tag{4}$$

$$\frac{\partial(\rho\overline{u}_i)}{\partial t} + \frac{\partial}{\partial x_j}\left(\rho\overline{u}_i\overline{u}_j + \rho\overline{u_i'u_j'}\right) = -\frac{\partial\overline{p}}{\partial x_i} + \frac{\partial\overline{\tau}_{ij}}{\partial x_j} \tag{5}$$

where $\overline{\tau}_{ij}$ are the components of the mean viscous stress tensor:

$$\overline{\tau}_{ij} = \mu\left(\frac{\partial\overline{u}_i}{\partial x_j} + \frac{\partial\overline{u}_j}{\partial x_i}\right).$$

The inclusion of the *Reynolds stresses* $\rho\overline{u_i'u_j'}$ into the conservation equations introduces new unknowns to the equation system. These cannot be expressed in terms of the known variables. To close the equation system these quantities need to be modeled using a turbulence model. The values for these coefficients are usually the result of a combination of theoretical considerations, computer optimisation and experimental measurements on simplified flows [27]. But even these empirical 'constants' have evolved in the 40 years of their existence, so it can be assumed that they have not yet reached generality for all possible flow configurations. For example the value for C_2 in the k-ε model (see 4.1.1) has changed by 4% from initially 2.0 to 1.92. Some research was done in a-priori parameter identification by Qian et al. [28], Bardow et al. [29], and others, but all these considerations did not lead to a better understanding of the impact of the parameters to the behaviour of the solution.

4.1.1 Standard k-ε Model

One way of modeling is to solve a transport equation for the rate of dissipation ε of turbulent kinetic energy k leading to the k-ε model first proposed by Jones and Launder [4] where $(\sqrt{k^3}/\varepsilon) \sim l$, with l being the turbulent length scale. The main problem of the k-ε model is its treatment in the near-wall region of the flow where the destruction-of-dissipation term is singular. To avoid this in a layer close to the wall the flow has to be treated seperately by a wall function. The resolution of the grid close to the walls has to be sufficiently fine for the wall functions to yield reasonable results, meaning additional care needs to be taken when solving a problem using this model. The equations as they are implemented in OpenFOAM are as follows:

$$
\begin{aligned}
\mu_t &= \rho C_\mu k^2/\varepsilon && (6)\\
\rho\frac{\partial k}{\partial t} + \rho U_i \frac{\partial k}{\partial x_i} &= \tau_{ij}\frac{\partial U_i}{\partial x_j} - \rho\varepsilon \\
&\quad + \frac{\partial}{\partial x_i}\left[(\mu + \mu_T/s_k)\frac{\partial k}{\partial x_i}\right] && (7)\\
\rho\frac{\partial \varepsilon}{\partial t} + \rho U_i \frac{\partial \varepsilon}{\partial x_i} &= C_1\frac{\varepsilon}{k}\tau_{ij}\frac{\partial U_i}{\partial x_j} - C_2\rho\frac{\varepsilon^2}{k} \\
&\quad + \frac{\partial}{\partial x_i}\left[(\mu + \mu_T/s_\varepsilon)\frac{\partial \varepsilon}{\partial x_i}\right] && (8)
\end{aligned}
$$

There are five arbitrary coefficients in this formulation, each influencing different aspects of the development of the flow. The commonly recognised standard values as implemented in the OpenFOAM CFD software are given in Table 1.

Table 1: Standard values for the k-ε model as implemented in OpenFOAM

s_k	s_ε	C_1	C_2	C_μ
1.0	1.3	1.44	1.92	0.09

4.1.2 Menter k-ω-SST Model

Another approach is to model the specific dissipation rate ω, as Wilcox [5] suggested in his version of the k-ω model, in which $(\sqrt{k}/\omega) \sim l$. Menter [30] introduced a modification to that model combining the near-wall treatment of the k-ε model and the accuracy in predicting the free flow from the k-ω model. He used blending functions to switch from one model to the other. The eddy viscosity equation is modified to account for the transport effects of the principle turbulent shear stress (hence the name k-ω-SST). Menter's formulation is widely used in aerodynamics and is a good example for the capability of the genetic optimisation as it contains no less than eleven arbitrary coefficients, of which the default values are given in Table 2. The implementation of this model in OpenFOAM uses the following equations:

$$
\begin{aligned}
\mu_t &= \frac{\rho a_1 k}{\max(a_1\omega, SF_2)} && (9)\\
\rho\frac{\partial k}{\partial t} + \rho U_i \frac{\partial k}{\partial x_i} &= \tilde{P}_k - \beta^*\rho k\omega \\
&\quad + \frac{\partial}{\partial x_i}\left[(\mu + s_k\mu_t)\frac{\partial k}{\partial x_i}\right] && (10)\\
\rho\frac{\partial \omega}{\partial t} + \rho U_i \frac{\partial \omega}{\partial x_i} &= \rho\frac{\gamma\tilde{P}_k}{\nu_t} - \beta\rho\omega^2 \\
&\quad + \frac{\partial}{\partial x_i}\left[(\mu + s_\omega\mu_t)\frac{\partial \omega}{\partial x_i}\right] \\
&\quad + 2(1-F_1)\frac{\rho s_{\omega 2}}{\omega}\frac{\partial k}{\partial x_i}\frac{\partial \omega}{\partial x_i} && (11)
\end{aligned}
$$

using a production limiter

$$
P_k = \mu_t\frac{\partial U_i}{\partial x_j}\left(\frac{\partial U_i}{\partial x_j} + \frac{\partial U_j}{\partial x_i}\right) \rightarrow \tilde{P}_k = \min\left(P_k, c_1\beta^*\rho\omega k\right).
$$

Each of the constants $\phi \epsilon \{\beta, \gamma, s_k, s_\omega\}$ is a blend of an inner ϕ_1 and outer ϕ_2 constant, blended via:

$$\phi = F_1\phi_1 + (1 - F_1)\phi_2$$

with blending function

$$F_1 = \tanh\left[\min\left[\max\left(\frac{\sqrt{k}}{\beta^*\omega y}, \frac{500\nu}{y^2\omega}\right), \frac{4\rho s_{\omega 2}k}{\mathrm{CD}_{k\omega}y^2}\right]^4\right]$$

$$\mathrm{CD}_{k\omega} = \max\left(2\rho s_{\omega 2}\frac{1}{\omega}\frac{\partial k}{\partial x_i}\frac{\partial \omega}{\partial x_i}, 10^{-10}\right)$$

$$F_2 = \tanh\left[\max\left(2\frac{\sqrt{k}}{\beta^*\omega y}, \frac{500\nu}{y^2\omega}\right)^2\right]$$

where ρ is the density, $\nu_t = \mu_t/\rho$ is the turbulent kinematic viscosity, μ is the molecular dynamic viscosity, y is the distance from the field point to the nearest wall. F1 is equal to zero away from the surface (k-ε model), and switches to one inside the boundary layer (k-ω model). Note that the production limiter coefficient c_1 is proposed as a constant in the original paper by Menter [30], but is implemented as a variable in OpenFOAM.

Table 2: Standard values for the k-ω-SST model in OpenFOAM

s_{k1}	$s_{\omega 1}$	γ_1	β_1
0.85034	0.5	0.5532	0.075
s_{k2}	$s_{\omega 2}$	γ_2	β_2
1.0	0.85616	0.4403	0.0828
a_1	c_1	β^*	
0.31	10	0.09	

4.1.3 Spalart-Allmaras 1-eqn model

The Spalart-Allmaras model was originally developed to model aerodynamic flows [3]. It is a one equation model as it only contains transport equation for the turbulent property $\tilde{\nu}$. The various terms in the formulation can be identified as diffusion, convection, production and destruction of this quantity. Each of these contributions to the transport equation has to be chosen carefully to account for the physics of the flow. Together with some dimensional analysis and modifications for the sake of numerical stability the original model by Spalart and Allmaras is most often used in this form:

$$\begin{aligned}\frac{\partial \tilde{\nu}}{\partial t} + u_j\frac{\partial \tilde{\nu}}{\partial x_j} &= C_{b1}(1 - f_{t2})\tilde{S}\tilde{\nu} - \left[C_{w1}f_w - \frac{C_{b1}}{\kappa^2}f_{t2}\right]\left(\frac{\tilde{\nu}}{d}\right)^2 \\ &+ \frac{1}{s}\left[\frac{\partial}{\partial x_j}\left((\nu + \tilde{\nu})\frac{\partial \tilde{\nu}}{\partial x_j}\right) + C_{b2}\frac{\partial \tilde{\nu}}{\partial x_i}\frac{\partial \tilde{\nu}}{\partial x_i}\right] \end{aligned} \tag{12}$$

with

$$\nu_t = \tilde{\nu}f_{v1}, \quad f_{v1} = \frac{\chi^3}{\chi^3 + C_{v1}^3}, \quad \chi := \frac{\tilde{\nu}}{\nu}$$

$$\tilde{S} \equiv S + \frac{\tilde{\nu}}{\kappa^2 d^2}f_{v2}, \quad f_{v2} = 1 - \frac{\chi}{1 + \chi f_{v1}}$$

and

$$
\begin{aligned}
S &\equiv \sqrt{2\Omega_{ij}\Omega_{ij}}, \quad \Omega_{ij} \equiv \frac{1}{2}(\frac{\partial u_i}{\partial x_j} - \frac{\partial u_j}{\partial x_i}) \\
f_w &= g\left[\frac{1 + C_{w3}^6}{g^6 + C_{w3}^6}\right]^{1/6}, \quad g = r + C_{w2}(r^6 - r) \\
r &\equiv \frac{\tilde{\nu}}{\tilde{S}\kappa^2 d^2} \\
f_{t2} &= C_{t3}\exp(-C_{t4}\chi^2)
\end{aligned}
$$

d is the distance to the closest wall. The trip f_{t2} was a numerical fix by Spalart and Allmaras that makes $\tilde{\nu} = 0$ a stable solution. This behaviour was desired in conjunction with a trip function $f_{t1}\Delta U$ given in the original reference to control the transition point in the flow. But according to Rumsey [31] most users do not employ this trip function, but run the model in fully turbulent mode. The proposed standard values for the coefficients are given in Table 3. The parameter C_{w1} is implicitly calculated from

$$C_{w1} = \frac{C_{b1}}{\kappa^2} + \frac{1 + C_{b2}}{s}.$$

Table 3: Standard values for the Spalart Allmaras model in OpenFOAM

C_{b1}	C_{b2}	s	κ	
0.1355	0.622	0.666	0.41	
C_{w2}	C_{w3}	C_{v1}	C_{t3}	C_{t4}
0.3	2	7.1	1.2	0.5
a_1	C_1	β^*		
0.31	10	0.09		

4.2 Parameter Identification

The above mentioned turbulence closure models include a number of parameters that need to be calibrated to the type of flow that is subject of the investigation. Surprisingly, most of these coefficients have little or no physical relevance at all and are merely empirical. The number of parameters vary from model to model with up to twelve in the Spalart-Allmaras closure. Lengthy experiments have to be conducted to estimate a range of values for the coefficients that best describes a specific type of flow covered by the experiment. Even though the authors of the models themselves provided standard values for the flows they investigated, these standards are used by industrial users regardless if they are fit to adequately describe a problem, or not. To identify those parts of the model equations that are most perceptible to variations in the coefficients a simple parameter identification study was performed where only one parameter at a time was changed while the others were held fixed at their standard values listed in tables 1, 2 and 3. The variation ranged from 60% to 140% around the standard value. This was neccessary to make sure the optimisation algorithm only optimised those values that have a real impact on the properties of the flow. Otherwise the method could not map unambigously between the value of a constant and the fitness of the solution and would therefore not converge.

5 Test Cases

5.1 Fitness Function

The most important aspect for a genetic optimisation algorithm to work is a proper definition of the fitness function. The return value of this function attributes the quality of a solution and the decision if a solution is fit for mating or will be discarded is based on this value. In the testcases described above we used experimental data and sampled data from the simulations and calculated the root mean square error between those two. This value is by definition always positive as a fitness value should be [10]. As the aim of the optimisation is to reach a solution as close to the experimental results as possible, we search to minimise the r.m.s. error. That means in reverse a smaller fitness function values represent better solutions. In case several sample regions are to be compared simultaneously, the errors are cumulated.

5.2 Backward-Facing Step

The flow over a backward-facing step is investigated and the results are compared to experimental data obtained by Makiola [32]. Simulations were done at two different Reynolds numbers $Re = 15,000$ and $Re = 64,000$ and both the standard k-ε and the k-ω turbulence model were applied. Computational meshes of varying sizes were used to ensure grid convergence and the final results are from grid independent solutions. To simulate one generation of fifty individuals takes approximately 21 minutes on ten computing cores in parallel. The algorithm saves time by not recalculating individuals that have been passed on from previous generations, so the total runtime for 30 generations is about seven hours.
The geometry of the case had an expansion ratio of $h/H = 2$ (see Figure 2) and the examined quantity was the normalised velocity u/U_0 at three different positions $x/H = 1, 3$ and 6 in the channel. That means the fitness was estimated as being the mean square root error between the simulated results and the velocity data measured in the experiment. The smaller the difference between the results, the better was the fitness of the solution. A parabolic velocity profile was prescribed at the inlet. Further only the results at position $x = 3H$ are shown exemplary for the complete dataset. That position is close to the center of the main recirculation vortex.

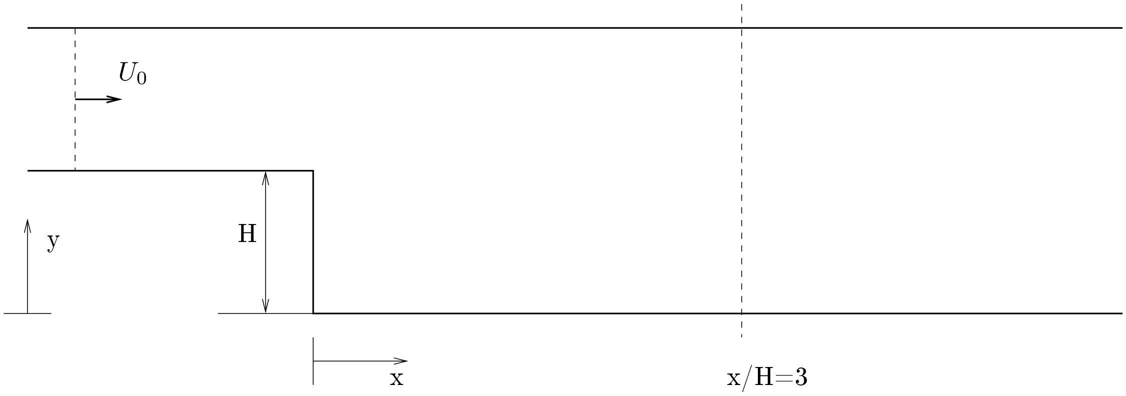


Figure 2: Geometry of the backward facing step test case

5.3 Conical Concentrator and Sudden Expansion

In order to assess the current state of the art in CFD modeling in a medical device model, Stewart et. al. [33] from the American Food & Drug Administration (FDA) designed a benchmark case to develop guidelines for CFD users in industry. Figure 3 shows the dimensions of the nozzle geometry used in the laboratories to obtain experimental results. The length of the inlet and outlet channels was not specified and should be chosen to ensure fully developed turbulent flow before entering the conical concentrator and the outflow condition should not influence the reattachment point in the model. In the simulation the length of the inlet and outlet channels were chosen as $15d$ and $300d$ respectively, with d being the diameter of the throat. For a throat Reynolds number of 5000, the inlet velocity was specified as $0.46m/s$. The best simulation results according to the authors were achieved using the Spalart-Allmaras one-equation turbulence model [3] (see also Section 4.1.3). Experimental data for this case was gathered from three different, independent laboratories and was made publicly available [34]. Since this case setup is slightly more complicated and requires a larger grid than the previous one, computation time was considerably longer. One set of 50 individuals over 30 generations took about 36 hours to complete on 10 cores.

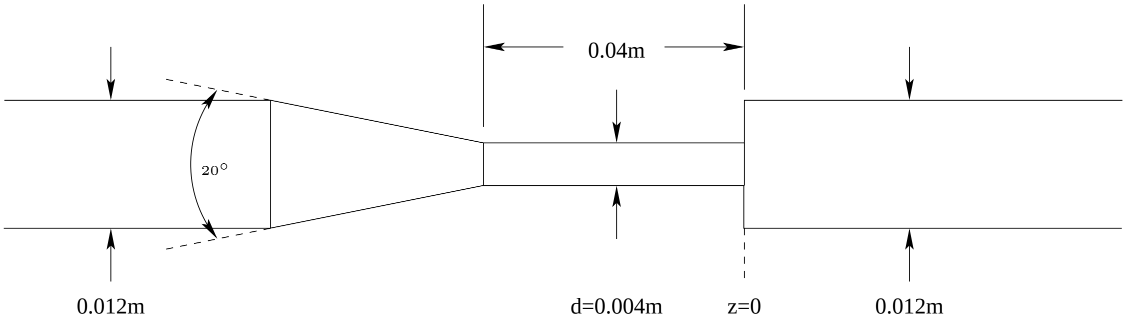


Figure 3: Dimensions of nozzle for the FDA test case

5.4 Influence of Coefficients

Due to the linear character of the coefficients in the models it is sufficient to show only the lower and upper bounds of the investigated range, since intermediate values all lie between these extremes. Tables 4 and 5 show fitness values relative to those obtained using the standard model coefficients. The lower bound for this investigation was 60% of the standard value, the upper bound was 140% of the standard value. Since smaller fitness values mean a better agreement with the experimental data, positive relative values represent an increase in the solution quality, while negative relative values stand for decreased quality. The relative fitness is estimated by evaluating

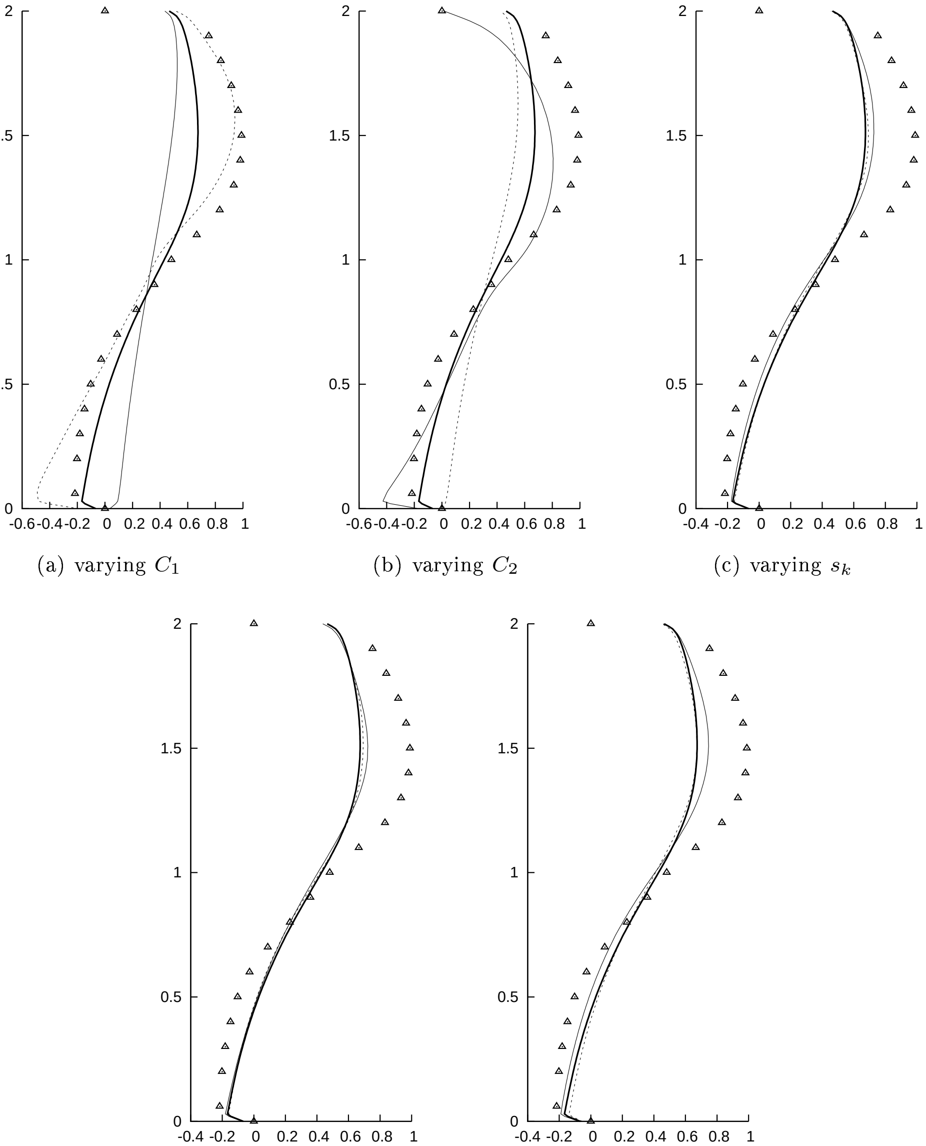
$$f_{rel} = \frac{f_{std} - f_{var}}{f_{std}} \tag{13}$$

where f_{std} is the fitness of the solution using the standard parameter values and f_{var} is the fitness value from the simulation with a modified parameter.
In the case of the k-ε model Fig. 4 shows that only two parameters have a significant influence on the calculated velocity profiles, namely C_1 and C_2. In physical terms these two parameters balance the production and dissipation of turbulent kinetic energy as the model Eqn. 8 shows. For the optimisation process using the genetic algorithm that means that only two instead of five variables need to be considered, significantly speeding up the convergence and accuracy of the process.

A similiar investigation of the coefficients of the k-ω-SST model was performed. Only a few of the parameters have a noticeable impact on the development of the flow in this particular case, so the optimisation will concentrate on these, which are namely $\gamma_1, \gamma_2, \beta_1, \beta_2$ and $\beta^\star$.

Table 4: Relative fitness values with varied coefficients for the k-ω-SST model. Positive values stand for improved solution quality compared to the standard coefficients while negative values mean decreased quality.

parameter	f_{rel}(60%)	f_{rel}(140%)
s_{k1}	0.010	0.007
s_{k2}	0.000	0.006
$s_{\omega 1}$	0.028	-0.020
$s_{\omega 2}$	0.013	-0.005
β_1	-0.139	0.097
β_2	-0.101	0.064
γ_1	0.069	-0.094
γ_2	0.059	-0.072
a_1	-0.218	0.036
c_1	-0.024	0.003
$\beta^\star$	0.199	-0.129



(a) varying C_1 (b) varying C_2 (c) varying s_k

(d) varying s_ε (e) varying C_μ

Figure 4: Variations of the coefficients in the k-ε model and their influence on the velocity profile at position $x/h = 3$ downstream of the step. The bold line is the result using the standard value from Table 1; the dotted line represents 140% of this value; the thin line is 60% of the standard value. The triangles mark experimental data by Makiola [32].

The result of the variation analysis for the second test case and the SpalartAllmaras model is shown in Table 5. In this case almost all the parameters need to be considered in the optimisation process.

5.5 Optimisation using GA

Based on the observations from the previous section, we now tried to find an optimal setting of the closure coefficients. The operators used were a single-point crossover operator, tournament selection and bitwise mutation. The parameters for the setup of the GA are a crossover probability P_C of 0.6, a mutation probability P_M of 0.03 and the population size S of 50 individuals. Analysis of the convergence showed no significant change to the optimal solution after just a few generations so the process was terminated

Table 5: Relative fitness values with varied coefficients for the SpalartAllmaras model. Positive values stand for improved solution quality compared to the standard coefficients while negative values mean decreased quality.

parameter	$f_{rel}(60\%)$	$f_{rel}(140\%)$
C_{b1}	0.210	-4.527
C_{b2}	0.206	-0.223
C_{v1}	-1.502	0.749
C_{v2}	-0.405	0.428
C_{w2}	0.094	-0.113
C_{w3}	-0.031	0.036
κ	0.036	-0.642
s	0.763	-0.735

after 30 iterations. The fitness of a solution is measured as deviation from experimental data. In case of the backward-facing step we calculate the sum of the root mean square errors of the velocity field at three positions in the flow downstream of the step. The smaller the error the better the fitness of an individual solution. Because of the non-deterministic character of the optimisation routine multiple performances of the algorithm do not always give the same results. A statistical analysis of a set of tests shows that all results lie within a standard deviation of 4%.

5.5.1 Backward-Facing Step

The estimated optimal values are listed in Table 6 for the k-ϵ model and in Table 7 for the k-ω model respectively. Fig. 5 shows the velocity profiles at different positions downstream as calculated using the optimised coefficients compared to the results obtained using the standard values included in OpenFOAM. The parabolic shape of the velocity profile is better captured by the optimised setup further downstream of the step. Using the standard coefficients the transition to fully developed channel flow takes place considerably faster, while the optimised profile maintains the dominance of the flow in the upper half of the channel in accordance to experiment.

Table 6: Optimum values and standard deviations for the k-ϵ model

C_1	$\sigma(C_1)$	C_2	$\sigma(C_2)$
1.91	0.082	1.86	0.093

Table 7: Optimum values and standard deviations for the k-ω-SST model

γ_1	$\sigma(\gamma_1)$	γ_2	$\sigma(\gamma_2)$
0.606	0.018	0.510	0.021

β_1	$\sigma(\beta_1)$	β_2	$\sigma(\beta_2)$	$\beta^\star$	$\sigma(\beta^\star)$
0.053	0.003	0.076	0.019	0.095	0.0008

Another interesting quantity to look at in the development of the flow behind the step is the length of the main recirculation eddy. The k-ε model is known to underestimate this model [?]. Responsible for this are the coefficients C_1 and C_2 in equation 7. They control the rate of production and dissipation of the turbulent quantities and should be well balanced to give a realistic picture of the energy distribution in the flow. From the equations one could deduce that a slightly higher influence of the production term and a slightly lower influence of the dissipation term would increase the size of the recirculation eddy. And that is indeed the result of the optimisation process. Table 8 compares the calculated reattachment lengths of the vertex with those obtained experimentally.

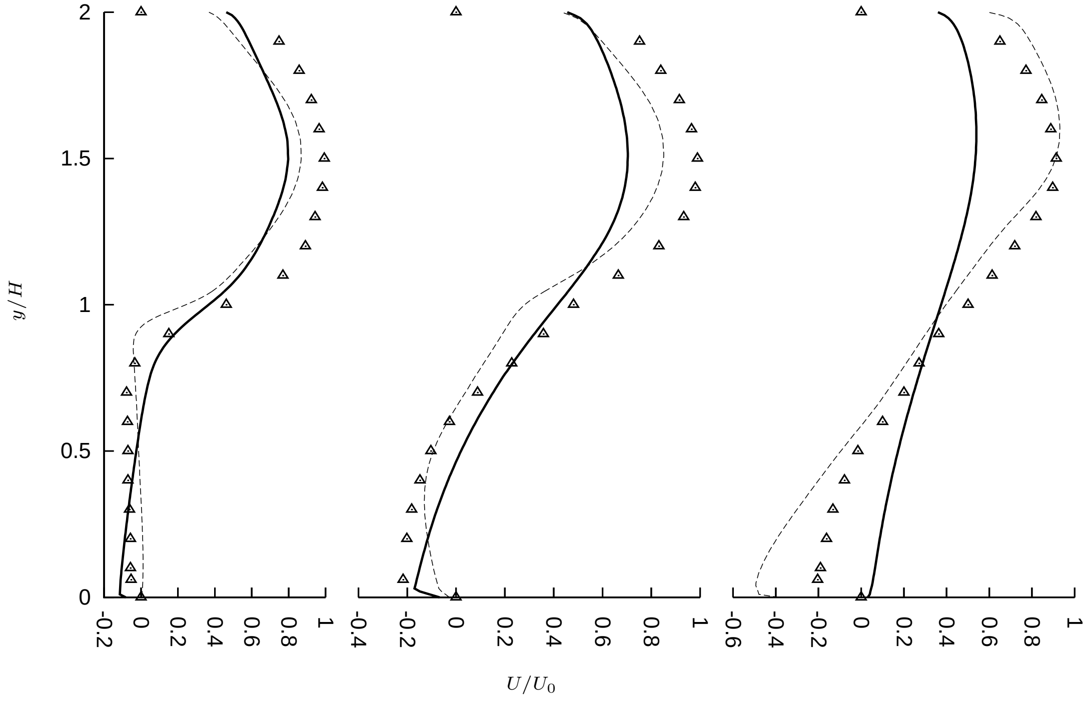


Figure 5: Velocity profiles at different positions downstream for the backward facing step case. From left to right: x/H = 3, 6, 10. Bold line: results using standard values for the k-ε model; dotted line: using optimised coefficients from Table 7. Triangles mark experimental data by Makiola [32].

Table 8: reattachment length of the main vortex normalized with step height for different case setups in comparison with experimental values [32]

		$k-\varepsilon$		$k-\omega$	
	exp.	std	opt	std	opt
Re=15k	8.2	4.7	11.6	5.6	9.8
Re=64k	8.6	4.8	10.9	6.7	11.1

5.5.2 Conical Concentrator and Sudden Expansion

The improved values for the four optimised coefficients in the SA model are listed in Table 9. As the model was developed for aerodynamic simulations, a boundary layer dominated flow as the conical concentrator is unaffected by most of the terms in the equation that control flow behaviour in the farfield. Therefore changing the values of the coefficients in these terms has no impact on the flow in the throat, but mainly on the recirculation area right after the sudden expansion. Taking that into account the fitness was evaluated at three different points behind the expansion, while the r.m.s. error in the concentrator or in the throat turned out to be largely invariant to changes in the model parameters. An improvement in performance compared to the standard model is yet noticeable. Especially in the zone behind the sudden expansion where $x/D > 0$, the algorithm was able to identify a set of parameters that almost perfectly matched the experimental data as can be seen in Figure 7.

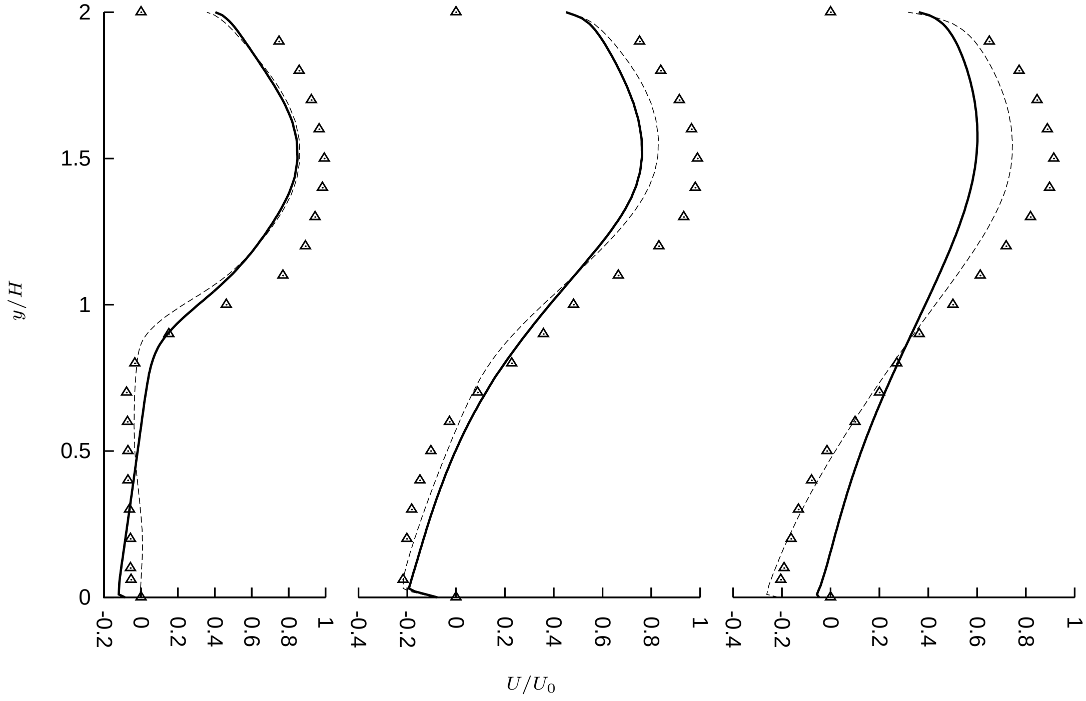


Figure 6: Velocity profiles at different positions downstream for the backward facing step case. From left to right: x/H = 3, 6, 10. Bold line: results using standard values for the k-ω-SST model; dotted line: using optimised coefficients from Table 7. Triangles mark experimental data by Makiola [32].

Table 9: Optimum values for the Spalart-Allmaras model

C_{b1}	C_{v1}	s	κ
0.172	9.187	0.447	0.274

6 Conclusions

Further tests need to be performed on more complicated flow regimes and different turbulence models. The results presented in this paper show the capability of genetic algorithms in combination with turbulence modelling. As an example two test cases were simulated, looking at three different, commonly used turbulence models. The presented results clearly show the ability of a non-deterministic evolution-based optimisation method to improve the accuracy of flow simulations. The technique can be used to find best-practise coefficients of popular turbulence models for a specific class of flow problems. With moderate effort it is now possible to identify the optimal setup for a simulation and to outperform the commonly used standard values. It still requires knowledge and understanding of the expected flow behaviour to classify a given problem and find an appropriate calibration case. It is also essential to have experimental or DNS data available to measure the fitness of a solution.

The results also show that this method is capable to expose the generality of a model in assessing the stability of a simulation to changes in the parameter space. If different variations of a case setup produce a wide variation of coefficients for one given model, that model seems not suitable to describe that kind of flow behaviour. The genetic algorithm can therefore be used to test new turbulence models on their versatility. In future work we will look into multi-objective optimisation to capture more features of the flow simultaneously. That will generate a broader picture of the influence of the parameters and will help to balance a solution between different if not contradictory quality requirements. Obviously for such an

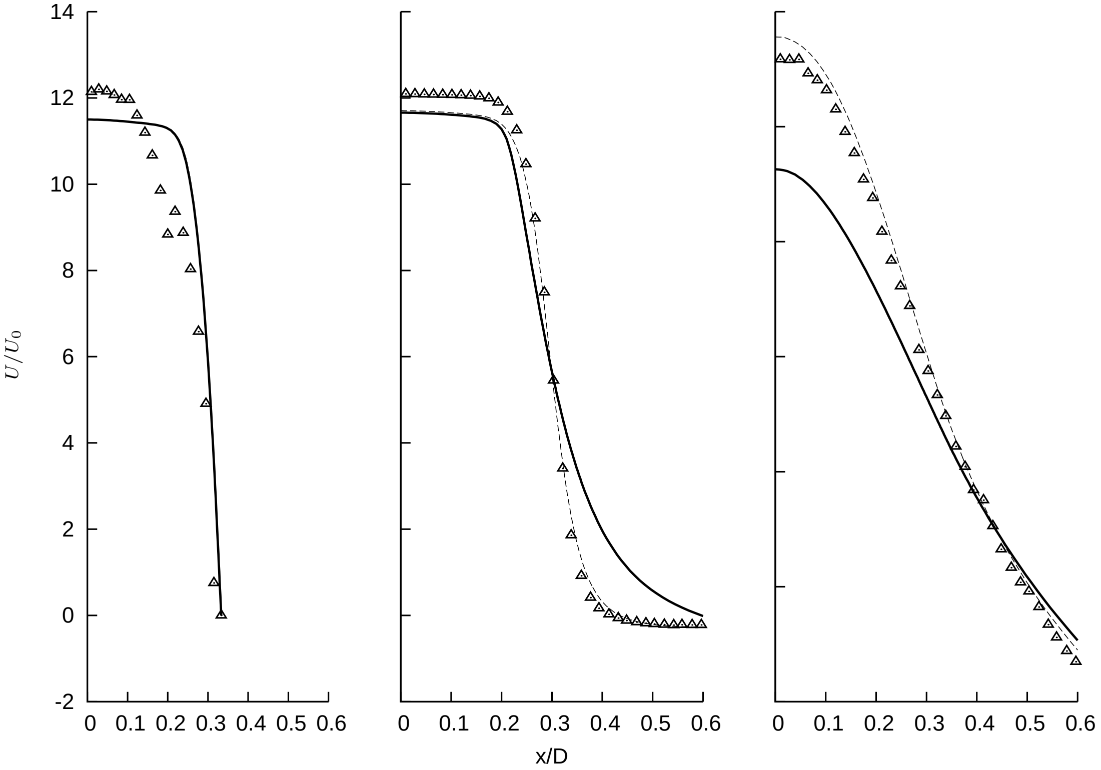


Figure 7: Velocity profiles at different positions of the nozzle for the FDA case. From left to right: x/D = -2, 2, 6. Bold line: results using standard values for the k-ε model; dotted line: using optimised coefficients from Table 9. Triangles mark experimental data taken from [34].

investigation, more detailed experimental or DNS data is required.

Acknowledgements

Björn Fabritius would like to thank the University of Exeter for providing funding for his PhD studies.

References

[1] P. Moin and K. Mahesh. Direct numerical simulation: A tool in turbulence research. *Annual Review of Fluid Mechanics*, 30:539âĂŞ78, 1998.

[2] E. Erturk, T.C. Corke, and C. Gökçöl. Numerical solutions of 2-D steady incompressible driven cavity flow at high reynolds numbers. *Int. J. Numer. Meth. Fluids*, 48:747–774, 2005.

[3] P. R. Spalart and S. R. Allmaras. A one-equation turbulence model for aerodynamic flows. *AIAA Paper*, 92-0439, 1992.

[4] W. P. Jones and B. E. Launder. The prediction of laminarization with a two-equation model of turbulence. *International Journal of Heat and Mass Transfer*, 15:301–314, 1972.

[5] D. C. Wilcox. *Turbulence Modeling for CFD.* DCW Industries, Inc., La Cañada, 3rd edition, 2006.

[6] H. J. Holland. *Adaptation in Natural and Artificial Systems.* University of Michigan Press, Ann Arbor, 1975.

[7] I. Rechenberg. *Evolutionsstrategie. Optimierung technischer Systeme nach Prinzipien der biologischen Evolution.* Frommann Holzboog, 1973.

[8] H.-P. Schwefel. *Evolution and Optimum Seeking.* Wiley & Sons, New York, 1994.

[9] L. J. Fogel, A. J. Walsh, and A. J. Owens. *Artificial Intelligence through simulated evolution.* John Wiley, New York, 1966.

[10] D. E. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning.* Addison-Wesley, 1989.

[11] K. A. deJong. *An Analysis of the Behavior of a Class of Genetic Adaptive Systems.* PhD thesis, University of Michigan, 1975.

[12] Xuesong Y., Wei W., Qingzhong L., Chengyu H., and Yuan Y. Designing electronic circuits by means of gene expression programming ii. In *Proceedings of the 7th international conference on Evolvable systems: from biology to hardware*, ICES'07, pages 319–330, Berlin, Heidelberg, 2007. Springer-Verlag.

[13] L. Jourdan, D. Corne, D. A. Savic, and G. A. Walters. LEMMO: Hybridising rule induction and NSGA II for multi-objective water systems design. In *CCWI2005, Computing and Control in the Water Industry*, pages 45–50, 2005.

[14] N. León-Rovira, E. Uresti, and W. Arcos. Fan shape optimisation using CFD and genetic algorithms for increasing the efficiency of electric motors. *Int. J. Computer Applications in Technology*, 30(1/2):47–58, 2007.

[15] R. Hilbert, G. Janiga, R. Baron, and D. Thèvenin. Multi-objective shape optimization of a heat exchanger using parallel genetic algorithms. *International Journal of Heat and Mass Transfer*, 49(15-16):2567 – 2577, 2006.

[16] Z. Michalewicz. *Genetic Algorithms + Data Structures = Evolution Programs.* Springer, Berlin, Heidelberg, 1996.

[17] N. Srinivas and K. Deb. Multiobjective optimization using nondominated sorting in genetic algorithms. *Evolutionary Computation*, 2(3):221–248, 1995.

[18] E. Zitzler, K. Deb, and L. Thiele. Comparison of multiobjective evolutionary algorithms: Empirical results. *Evolutionary Computation*, 8:173–195, 2000.

[19] C. M. Fonseca and P. J. Fleming. An overview of evolutionary algorithms in multiobjective optimization. *Evolutionary Computation*, 3:1–16, 1995.

[20] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: NSGA-II. *Evolutionary Computation, IEEE Transactions on*, 6(2):182 –197, 2002.

[21] K. Behzadian, Z. Kapelan, D. Savic, and A. Ardeshir. Stochastic sampling design using a multi-objective genetic algorithm and adaptive neural networks. *Environmental Modelling & Software*, 24(4):530 – 541, 2009.

[22] L. Jourdan, D. Corne, D. Savic, and G. Walters. Preliminary investigation of the 'learnable evolution model' for faster/better multiobjective water systems design. In C. A. Coello Coello et al., editor, *EMO 2005*, volume LNCS 3410, pages 841–855. Springer-Verlag Berlin Heidelberg, 2005.

[23] Message Passing Interface Forum. *MPI: A Message-Passing Interface Standard, Version 2.2.* High-Performance Computing Center Stuttgart, 2009.

[24] Ch. Gagné and M. Parizeau. Genericity in evolutionary computation software tools: Principles and case-study. *International Journal on Artificial Intelligence Tools*, 15(2):173–194, 2006.

[25] E. Gamma, R. Helm, R. Johnson, and J. Vlissides. *Design Patterns - Elements of Reusable Object-Oriented Software.* Addison Wesley, 1995.

[26] J. H. Ferziger and M. Perić. *Computational Methods for Fluid Dynamics.* Springer-Verlag, Berlin, 3rd rev. edition, 2002.

[27] K. Hanjalić and B.E. Launder. A reynolds stress model of turbulence and its application to thin shear flow. *Journal of Fluid Mechanics*, 52:609–638, 1972.

[28] Qian W. and Cai J. Parameter estimation of engineering turbulence model. *Acta Mechanica Sinica*, 17(4):302–309, 2001.

[29] A. Bardow, Ch. H. Bischof, H. M. Bucker, G. Dietze, R. Kneer, A. Leefken, W. Marquardt, U. Renz, and E. Slusanschi. Sensitivity-based analysis of the k-ϵ model for the turbulent flow between two plates. *Chemical Engineering Science*, 63(19):4763 – 4775, 2008.

[30] F. R. Menter, M. Kuntz, and R. Langtry. *Ten Years of Industrial Experience with the SST Turbulence Model.* Turbulence, Heat and Mass Transfer 4. Begell House, Inc, 2003.

[31] C. L. Rumsey. Apparent transition behavior of widely-used turbulence models. *International Journal of Heat and Fluid Flow*, 28:1460–1471, 2007.

[32] B. Makiola. *Experimentelle Untersuchungen zur Strömung über die schräge Stufe.* PhD thesis, Institut

für Hydromechanik, Universität Karlsruhe, 1992.

[33] P. Hariharan, M. Giarra, V. Reddy, and S. W. Day. Multilaboratory particle image velocimetry analysis of the FDA benchmark nozzle model to support validation of computational fluid dynamics simulations. *Journal of Biomechanical Engineering*, 133:410021–14, 2011.

[34] S. Stewart, E. Paterson, G. Burgreen, P. Hariharan, M. Giarra, V. Reddy, S. Day, K. Manning, S. Deutsch, M. Berman, M. Myers, and R. Malinauskas. Assessment of CFD performance in simulations of an idealized medical device: Results of FDA's first computational interlaboratory study. *Cardiovascular Engineering and Technology*, pages 1–22, 2012.