SHAPE OPTIMISATION USING COMPUTATIONAL FLUID DYNAMICS AND EVOLUTIONARY ALGORITHMS

STEVEN DANIELS[1], ALMA RAHAT, GAVIN TABOR, JONATHAN FIELDSEND, RICHARD EVERSON
[1]*College of Engineering, Mathematics & Physical Sciences, University of Exeter, S.Daniels@exeter.ac.uk*

Keywords: *Shape optimisation, Evolutionary Algorithm, sub-division curves, PyFoam, snappyHexMesh.*

Optimisation of designs using Computational Fluid Dynamics (CFD) are frequently performed across many fields of research, such as the optimisation of an aircraft wing to reduce drag, or to increase the efficiency of a heat exchanger. General optimisation strategies involves altering design variables with a view to improve appropriate objective function(s). Often the objective function(s) are non-linear and multi-modal, and thus polynomial time algorithms for solving such problems may not be available. In such cases, applying Evolutionary Algorithms (EAs - a class of stochastic global optimisation techniques inspired from natural evolution) may locate good solutions within a practical time frame. The traditional CFD design optimisation process is often based on a 'trial-and-error type approach. Starting from an initial geometry, Computational Aided Design changes are introduced manually based on results from a limited number of design iterations and CFD analyses. The process is usually complex, time-consuming and relies heavily on engineering experience, thus making the overall design procedure inconsistent, i.e. different 'best' solutions are obtained from different designers.

From the limitations of optimisation by-hand, using EAs and CFD simulations may be an attractive alternative. There have been other attempts to combine EAs and CFD simulation to optimise designs, see for example [1]. Although EAs do not guarantee locating the optimal solution, they may achieve good solutions consistently. In this present work, an automated framework combining python's EA library DEAP with OpenFOAM 2.3.1 was used. The communication of the python libraries with OpenFOAM was achieved using the pyFoam. PyFoam was used as an interface to control the OpenFOAM case set-ups for each proposed solution from the EA code and to post-process the data generated after each CFD calculation.

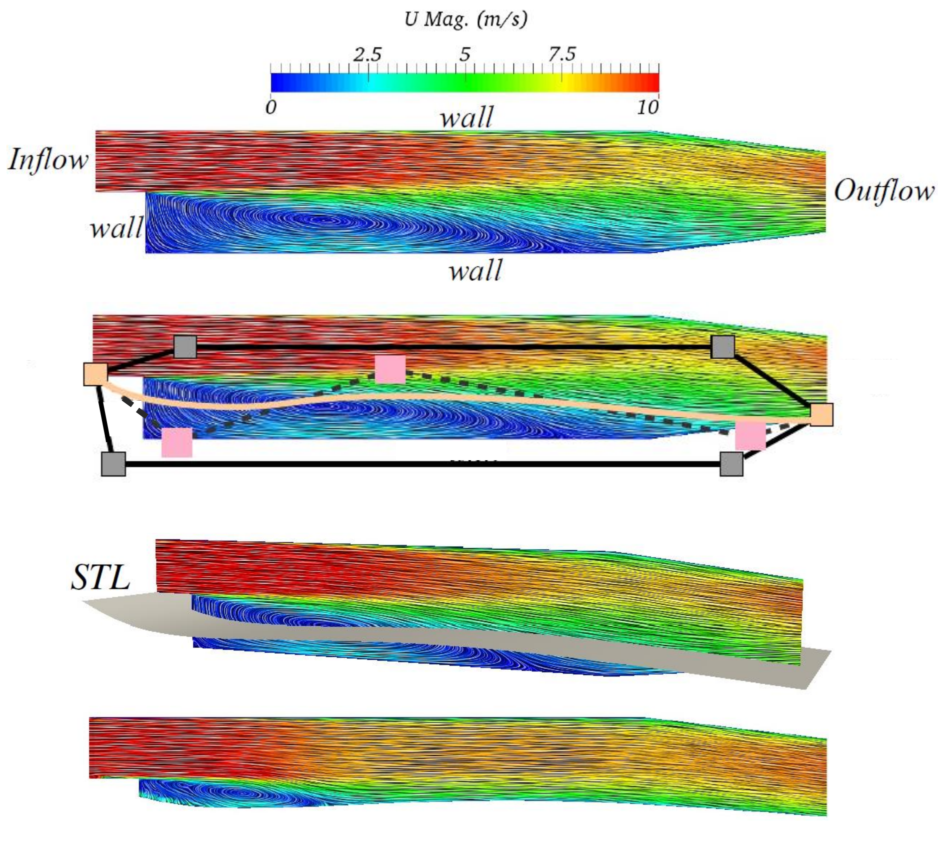

Figure 1: Shape optimisation procedure of the PitzDaily tutorial case.

While the aim of this project is to focus on a complex (potentially multi-objective) case, for the purpose of this abstract, a single-objective optimisation of the PitzDaily tutorial in OpenFOAM was used to demonstrate the performance of the proposed procedure. The chosen cost function for this case was to minimise the pressure-drop between the inflow and outflow boundary conditions, i.e.

$$f_1 = min(|p_i - p_o|), \tag{1}$$

where both pressures were obtained by averaging over the boundary condition faces. Figure 1 (top-to-bottom) shows the procedure in which the shape is altered after each evaluation. To change the geometry, sub-division curves were generated to form a new wall. The fixed points of the spline (indicated in orange squares) were placed across bottom boundary condition ('lowerWall'). The squares in pink indicate the control points that ma be altered by the EA, and thus change the curvature of the spline. After the new positions of the pink squares is proposed by the EA, a Stereolithography (STL) file is generated and passed to OpenFOAM. Using this file, the meshing utility snappyHexMesh was used to cut the domain away (and if required re-mesh the altered region). Subsequently, the case was run using the steady-state solver simpleFoam. After this, the cost function was obtained and the EA determines a new position for the coordinates of the pink squares.

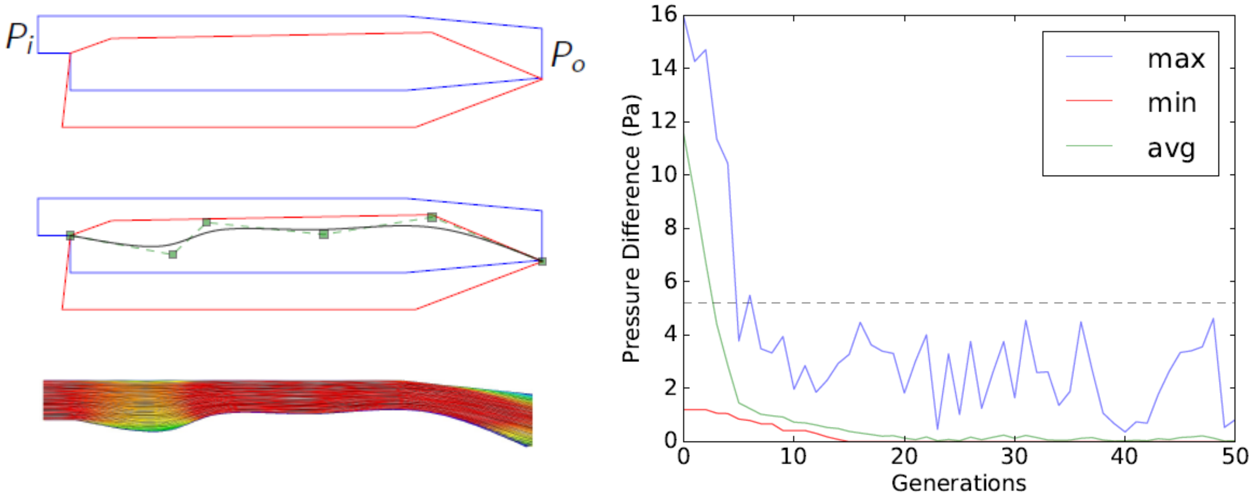

Figure 2: Single-Objective Optimisation of the PitzDaily tutorial case in reducing the pressure drop.

A single-objective optimisation run of the PitzDaily tutorial was performed with a basic real valued Genetic Algorithm ([2]). For the CFD calculation, the residual tolerances for the SIMPLE algorithm for velocity and pressure was set to 10^{-5} and 10^{-6} respectively. The pressure drop calculated from the original (base) case was measured as $|\Delta p| = 5.22$Pa. Figure 2 (left) shows the optimised shape of the PitzDaily geometry. The resulting pressure drop was calculated as $|\Delta p| = 4.7e^{-6}$Pa. The schematic diagrams above the contour diagram of velocity magnitude indicate the position of the control points, and the outer boundary region (in red) in which they must not exceed. The GA maintains a population of solutions: Figure 2 shows the maximum (worst), the average and the minimum (best) pressure drop across the population with each generation. The grey-dashed line indicates the pressure drop for the original PitzDaily geometry. It can be seen that the reduction of the best pressure difference between the inflow and the outflow is reached relatively quickly after 15 generations, and an improvement of the pressure-difference from the original case is achieved sooner than this.

Preliminary results of a recently proposed automated framework are presented here. While at present the representation of the new geometry is achieved by snappyHexMesh, other methods are currently under investigation for comparison. Furthermore, in the future, to allow for a comprehensive investigation into the performance of EAs in shape optimisation, we plan to investigate a number of test problems similar to those commonly observed in industry, such as airflow ducts and heat exchangers.

Acknowledgments

This work was supported by the UK Engineering and Physical Sciences Research Council (EPSRC) grant (reference number: EP/M017915/1) for the University of Exeter's College of Engineering, Mathematics, and Physical Sciences.

References

[1] A. Caputo, P. Pelagagge, and P. Salini, *Heat exchanger design based on economic optimisation.* Applied Thermal Engineering, 2008, vol. 28, pp. 1151–1159.

[2] T. Bäck, D. B. Fogel, and Z. Michalewicz, *Evolutionary computation 1: Basic algorithms and operators.* CRC Press, 2000, vol. 1.