V European Conference on Computational Fluid Dynamics
ECCOMAS CFD 2010
J. C. F. Pereira and A. Sequeira (Eds)
Lisbon, Portugal, 14-17 June 2010

NUMERICAL SIMULATION OF COOLING GAS INJECTION USING ADAPTIVE MULTISCALE TECHNIQUES

Wolfgang Dahmen[†],**Thomas Gotzen**[*] **and Siegfried Müller**[††]

[*]Institut für Geometrie und Praktische Mathematik, RWTH Aachen
Templergraben. 55, 52056 Aachen,
e-mail: gotzen@igpm.rwth-aachen.de
[††] e-mail: {dahmen, mueller}@igpm.rwth-aachen.de

Key words: Finite Volume Method, Film cooling, Cooling gas injection, Multiscale techniques, Grid adaptation

Abstract. *The interaction of a jet of cooling gas injected through single boreholes with a supersonic flow field plays an essential role in the design of innovative cooling systems. The detailed knowledge of this process will be helpful to derive effective boundary conditions for a homogenized mass flux of cooling gas through porous media.*

In a first step, a simplified two-dimensional configuration mimicking the gas injection through a slot of infinite length in a plate is investigated. These numerical results are thoroughly validated by van Driest's similarity solution for laminar boundary layers, computations performed with a non-adaptive code and experimental data provided by the Shock Wave Laboratory of the RWTH Aachen.

The numerical investigations verify that the multiscale-based grid adaptation concept is a reliable and efficient tool by which all physically relevant effects are automatically detected and appropriately resolved.

1 Introduction

In combustion chambers of rocket engines the walls are exposed to high temperatures. In order to avoid material damage effective cooling concepts are necessary. Here film cooling is an innovative concept to reduce the heat load of the wall that is subject of current research activities. However, there is still a severe lack of understanding concerning the fundamentals of the flow field and the thermal interaction.

The basic idea of film cooling is to inject cooling gas through boreholes in the wall surface such that a thin film develops at the wall. Thus, the high temperature gas in the burning chamber does not come into direct contact with the surface and the heat load at the wall will be reduced. To reliably produce such a thin cooling layer, the cooling gas has to be injected through many densely distributed holes. One option is the use of porous media. To avoid the resolution of corresponding fine scale geometric structures one may resort to upscaling strategies, for instance, based on local fine scale problems involving one or a few injection holes. Appropriate materials are being investigated.

The effectivity of such cooling concepts crucially relies on the comprehension of the mass flow through the porous media and the boreholes, respectively, and the interaction of the cooling gas with the attached high-speed flow field. A direct numerical simulation (DNS) that adequately resolves all geometric and corresponding flow scales for many injection holes or porous surfaces is far beyond the capacity of present and near future computer technology. Therefore the mass flux has to be modeled by homogenization and multiscale techniques. For this purpose, our investigations are firstly confined to the injection of cooling gas through single boreholes. The resulting insights are then used to determine effective boundary conditions for a homogenized formulation of the active cooling process with a multitude of boreholes.

In the present work we focus on the numerical simulation of cooling gas injected into a laminar, supersonic flow field through a single slot in a plate. The main purpose is (i) to extend the flow solver Quadflow[1] basically developed and applied to subsonic and transonic flow fields around airfoils[2] to such applications and (ii) to thoroughly validate the numerical results by analytical results[3], computations by Heufer[4] and by experimental data at the Shock Wave Laboratory of the RWTH Aachen[5].

Thus this paper is structured as follows: First, the basic concepts of the flow solver Quadflow concerning grid adaptation, grid generation and finite volume discretization are briefly summarized in Section 2. In Section 3.1, we then apply the solver to a supersonic flow field over a flat plate. This allows us to validate the results by van Driest's similarity solution. Here, in particular, the reliability and efficiency of the grid adaptation is of interest. Finally, in Section 3.2, we present computations where a cooling gas is injected by one slot in the plate and investigate the interaction with the laminar boundary layer and the supersonic flow field. We conclude with a summary of the main results and give an outlook to future work.

2 Numerical Method

The computations are performed using the adaptive flow solver Quadflow[1] . This solver has been developed over a period of more than one decade within the collaborative research center SFB 401 *Modulation of Flow and Fluid-Structure Interaction at Airplane Wings*[2]. It solves the Euler and Navier-Stokes equations around complex aerodynamic configurations by a cell-centered finite volume method on locally refined grids. Mesh adaptation is based on multiscale analysis. The computational grids are represented by block-structured parametric B-Spline patches.

The central objective is to realize adaptively generated discretizations that are able to resolve the physically relevant phenomena at the expense of possibly few degrees of freedom and correspondingly reduced storage demands. This requires a careful coordination of the core ingredients namely the discretization of the underlying system of partial differential equations, the generation and management of suitable meshes and the adaptation mechanisms.

Adaptation[6]: The main distinction from previous work in this regard lies in the fact that we employ here recent *multiresolution techniques*. The starting point is to transform the arrays of cell averages associated with any given finite volume discretization into a different format that reveals insight into the local behavior of the solution. The cell averages on a given highest level of resolution $l = L$ are represented as cell averages on some coarse level $l = 0$ where the fine scale information is encoded in arrays of *detail coefficients* of ascending resolution $l = 0, \ldots, L-1$. This requires a *hierarchy of meshes*. The multiscale representation is used to create *locally refined meshes*. Thus a principal objective is to extract the inherent complexity of the problem by placing as few degrees of freedom so as to still capture the features of the searched for solution within a given tolerance. A central mathematical problem is then to show that the essential information to be propagated in time is still kept with sufficient accuracy when working on locally coarser meshes.

Mesh Generation[7]: The adaptation strategy gives rise to locally refined meshes of quadtree respectively octree type. The second important ingredient is the generation of such meshes along with the information needed by the flow solver at any stage of a dynamical calculation. A key idea is to represent such meshes with as few parameters as possible while further successive refinements can be efficiently computed based on the knowledge of these parameters. This seems to be of vital importance with regard to (geometrically) non–stationary processes. Roughly speaking the mesh in each block results from evaluating a parametric mapping from the computational domain into the physical domain. Such mappings can be based on B-spline representations in combination with well established concepts from CAGD (computer aided geometric design). The quantities to be updated in time are the relatively few control parameters in those parametric representations, while mesh points on any level of resolution can be efficiently computed due to the locality of the B-spline representation. The fact that one needs indeed only

relatively few control points in order to generate meshes of good quality is partly due to the variation diminishing property of B-splines.

Flow Solver[8]: The discretization scheme has to meet the requirements of the adaptation concept and has to fit well with the mesh generation. This requires the development of a finite volume scheme for fairly general cell partitions that can cope, in particular, with hanging nodes and possible unstructured parts in complicated regions of the flow domain. For this purpose, the locally adapted grid is treated as a fully unstructured mesh with arbitrary polygonal/polyhedral control volumes in two and three space dimensions, respectively. The convective fluxes are determined by solving quasi–one–dimensional Riemann problems at the cell interfaces. Several approximate Riemann solvers (Roe, HLLC, AUSMDV) and upwind schemes (van Leer) have been incorporated. A linear, multidimensional reconstruction of the conservative variables is applied to increase the spatial accuracy. In order to avoid oscillations in the vicinity of local extrema and discontinuities, limiters with TVD property are used. Concerning the computation of the viscous fluxes, the gradients of the variables at cell interfaces are determined using the divergence theorem. Finally, the time–integration is performed by an explicit multistage Runge–Kutta scheme and a fully implicit Newton–Krylov type method, respectively.

3 Numerical results

The flow solver Quadflow has been primarily developed to investigate transonic flow fields around airfoils. In order to perform numerical simulations of cooling gas injection into a supersonic flow field, we need to validate the solver thoroughly by means of classical benchmark problems. For this purpose, we will first consider in Section 3.1 the supersonic flow over a 2D flat plate and compare the results with similarity solutions of van Driest[3]. The validated solver is then applied to the simulation of cooling gas injection in Section 3.2. The results are compared with experiments that were carried out at the Shock Wave Laboratory of the RWTH Aachen and numerical simulations provided by a non-adaptive code[5].

3.1 Supersonic flow over flat plate

Computational Setup: First of all, we consider a supersonic flow over a flat plate of length $150\,mm$, where the plate thickness in the computation is assumed to be zero. The free stream conditions at the leading edge of the plate are characterized by the Mach number $Ma_e = 2.6$, the Reynolds number $Re_e = 4.3 \cdot 10^6$ and the temperature $T_e = 488\,K$. These will also be used later on in Section 3.2 where in addition a cooling gas is injected through a slot in the plate. Since the flow field is supposed to be laminar, the computations are performed in two space dimensions only.

The flow field is characterized by a laminar boundary layer and an isotropic compression wave emanating at the leading edge of the plate. These physical effects have to be adequately resolved by locally anisotropic and isotropic grids, respectively. To account for

both opposing requirements, we therefore split the computational domain into basically two blocks $\Omega_B = [-0.01\,m, 0.2\,m] \times [0\,m, \delta]$ and $\Omega_F = [-0.01\,m, 0.2\,m] \times [\delta, 0.15\,m]$ for the offset at the plate and the far field, respectively, where $\delta = 0.005\,m$ is about four times the analytical prediction of the boundary layer thickness at the end of the plate according to van Driest[3].

Due to the high Reynolds number the boundary layer is very thin. To resolve this thin layer by dyadic, isotropic grid refinement requires a very high number of refinement levels resulting in a tremendously high number of cells. This can be avoided to some extent by concentrating grid lines at the wall resulting in stretched cells. On the other hand, the leading edge of the boundary layer needs to be resolved locally by an isotropic grid to capture the compression wave emanating at this point. To compensate for the stretching in normal wall direction, we also have to narrow grid lines in tangential direction at the leading edge. In addition, it turned out that the flow field in front of the plate has to be discretized as well to resolve the physical effects at the leading edge. Therefore the offset block Ω_B is split into two sub-blocks $\Omega_{B,1} = [-0.01\,m, 0\,m] \times [0\,m, \delta]$ in front of the plate and $\Omega_{B,2} = [0\,m, 0.2\,m] \times [0\,m, \delta]$ at the plate. In each block a B-spline mapping is constructed. In order to concentrate grid lines at the wall and at the leading edge of the plate ($x = 0\,m$), we apply stretching functions to the parameter spaces of the B-Spline functions corresponding to the tangential and wall-normal direction, respectively. In the far field, i.e., block Ω_F, a Cartesian mesh is used.

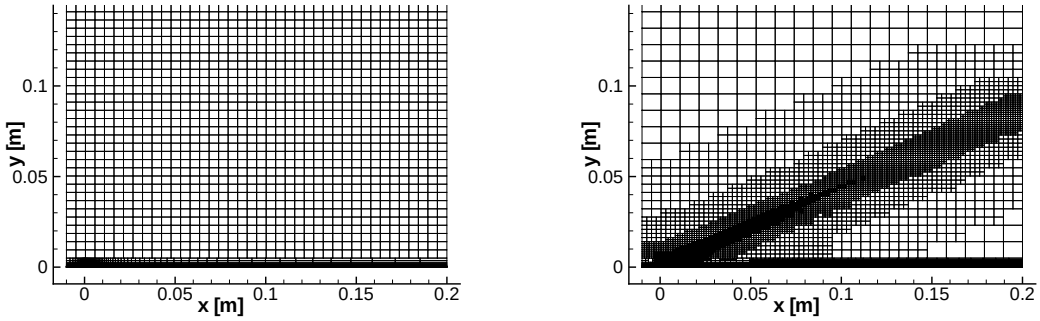
In front of the flat plate, we impose symmetric boundary conditions, i.e.,

$$\frac{\partial \rho}{\partial n} = 0,\ \frac{\partial u}{\partial n} = 0,\ v = 0,\ \frac{\partial p}{\partial n} = 0, \tag{1}$$

for density ρ, velocity components in x- and y-direction u and v, respectively, and pressure p. On the surface of the plate, we prescribe no slip conditions and a constant wall temperature of $T_w = 293\,K$ modeling an impermeable and isothermal wall.

The coarse grid discretization corresponding to level $l = 0$ is composed of 605 cells. Note that the current implementation of the no-slip conditions requires to resolve all cells attached to the wall by the highest refinement level in order to avoid that hanging nodes occur in these cells, i.e., grid adaptation is prohibited in these cells. Due to grading, the resolution may decrease gradually in wall-normal direction. This can be seen in Fig. 1(a). Since the flow field is initialized by the homogeneous free stream conditions, the computation is started on the first uniformly refined refinement level $l = 1$, see Fig. 1(a). During the computation grid adaptation is performed whenever the averaged density residual has dropped by 4 orders of magnitude for five adaptations. Afterwards, computation continues until a residual drop of 10^{-5} is reached. Due to the steady state character of the solution, the time discretization is performed implicitly by a backward Euler step and the time step is chosen locally with respect to a varying global CFL number determined by the following CFL evolution strategy

$$CFL_{k+1} = min(CFL_{min} \cdot 1.05^k, CFL_{max}), \tag{2}$$



(a) (b)

Figure 1: Initial grid (uniform 1st refinement level) (a) and final grid (b).

where CFL_{min} and CFL_{max} are set to 1 and 20, respectively. Note that the index k enumerates the number of time steps since the last adaptation, i.e., after each adaptation the CFL number is again set to CFL_{min}. For solving the Riemann problem at the cell interfaces we use the AUSMDV solver. The viscosity is calculated by the Sutherland Law

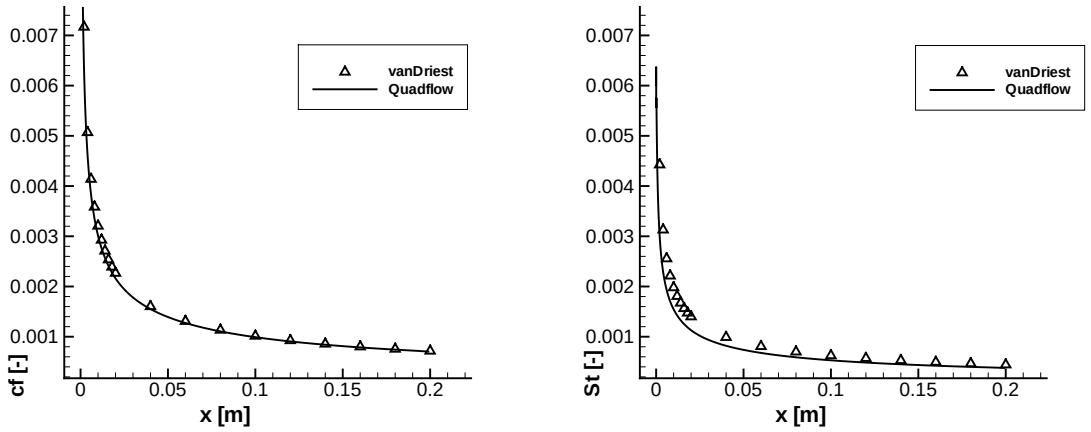
$$\mu(T) = \mu_e \left(\frac{T}{T_e}\right)^{3/2} \frac{T_e + S}{T + S} \tag{3}$$

with the Sutherland constant $S = 110\,K$ and the Prandtl number is set to $Pr = 0.72$. The wall-temperature ratio is given as $T_w/T_e = 1.665$.

Numerical Results: Starting from a homogeneous flow field, the initial grid, see Fig. 1(a), is locally refined during the computation by means of multiscale-based grid adaptation. The final grid corresponding to the steady state solution consists of about 45000 cells distributed over 5 levels of refinement, see Fig. 1(b). For comparison, the uniformly refined mesh corresponding to $L = 5$ consists of about 620000 cells. Hence the computational complexity is reduced by a factor of about 14.

As can be concluded from Fig. 1(b), the compression wave has been automatically detected during runtime by performing grid adaptation. At the trailing edge, see Fig. 4(b), the compression wave is adequately resolved by cells on the highest refinement level. In the far field, since the compression becomes weaker, it no longer needs to be resolved on the highest refinement level.

For the boundary layer we have similar conclusions. Although the grid is a priorily refined in this layer, due to the stretching applied to the parameter spaces of the B-Spline mappings, the grid is further refined. This is verified by Fig. 3, where the resolution of the boundary layer at the end of the plate is shown for the initial and the final grid, respectively. The numerical boundary layer thickness at the end of the plate $\delta_{num} = 1.12\,mm$ that can be depicted from Fig. 3(b) fits very well with the analytical prediction $\delta = 1.14\,mm$ of van Driest's similarity solution. The boundary layer is, at the end of the plate, resolved by about 50 cells in wall-normal direction.



(a) (b)

Figure 2: Supersonic flow over a flat plate compared with similarity solution according to van Driest: skin friction (a) and Stanton number (b).

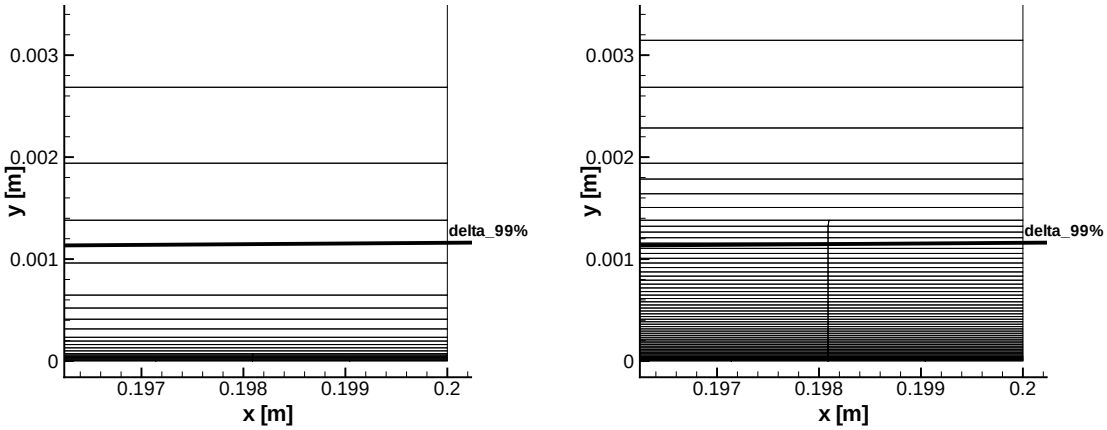
It has to be emphasized that the concept of multiscale-based grid adaptation reliably and automatically detects all physical relevant effects, namely, the compression wave and the boundary layer.

The results obtained are validated by similarity solutions of van Driest[3]. As can be seen in Fig. 2, the numerical results for the skin friction fits very well with the theoretical prediction. The results for Stanton number, describing the heat flux at the wall, show slight differences, but the quality is still satisfying. Regarding the cooling gas injection, this indicates that a good agreement for the skin friction does not necessarily ensure reliable results for the cooling efficiency.

3.2 Cooling gas injection in laminar flow

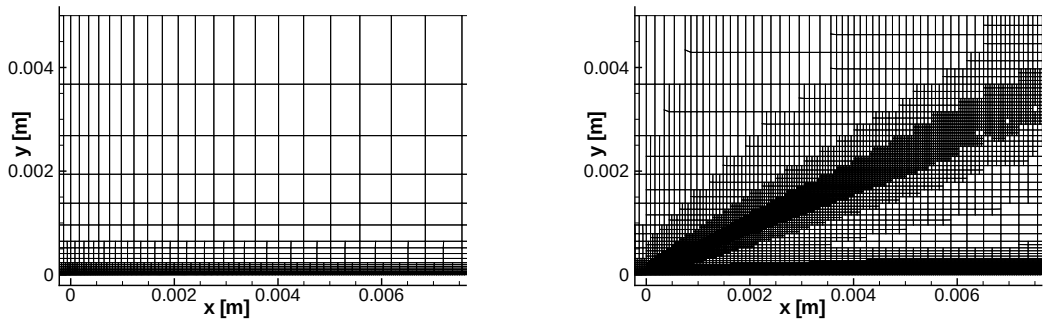
Experimental Setup: At the Shock Wave Laboratory of the RWTH Aachen experiments have been performed at supersonic inflow conditions, where the main flow field hits a wedge with edge length 150 mm at an angle of attack $\alpha = 30°$. Typically an oblique shock is forming at the leading edge of the wedge. Cooling gas, here air, is injected through a slit in the surface that is located 55 mm from the leading edge of the wedge, see Fig. 5. Characteristic parameters of the experimental setup are the slit width L_s, the injection angle Θ and the blowing ratio $F = \rho_c u_c / \rho_e u_e$, where ρ and u are the density and the velocity, and the subscripts e and c denote the respective values behind the oblique shock wave and the condition of the injected cooling gas. These parameters can be varied as follows: $L_s \in [0.5\,mm, 1\,mm]$, $\Theta \in [30°, 90°]$ and $F \in [0.0151, 0.065]$. Note that $\Theta = 90°$ corresponds to an injection orthogonal to the wall.

Computational Setup: The flow field in the experiment is essentially laminar and,



(a) (b)

Figure 3: Resolution of the boundary layer: initial grid (uniform 1st refinement level) (a) and final grid (b).



(a) (b)

Figure 4: Resolution of the leading edge of the plate: initial grid (uniform 1st refinement level) (a) and final grid (b).

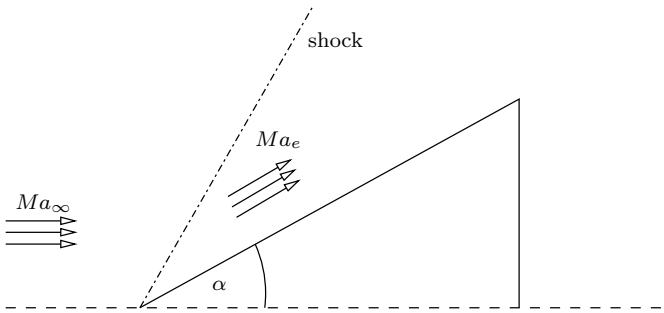


Figure 5: Side view of experimental setup: upper part of wedge.

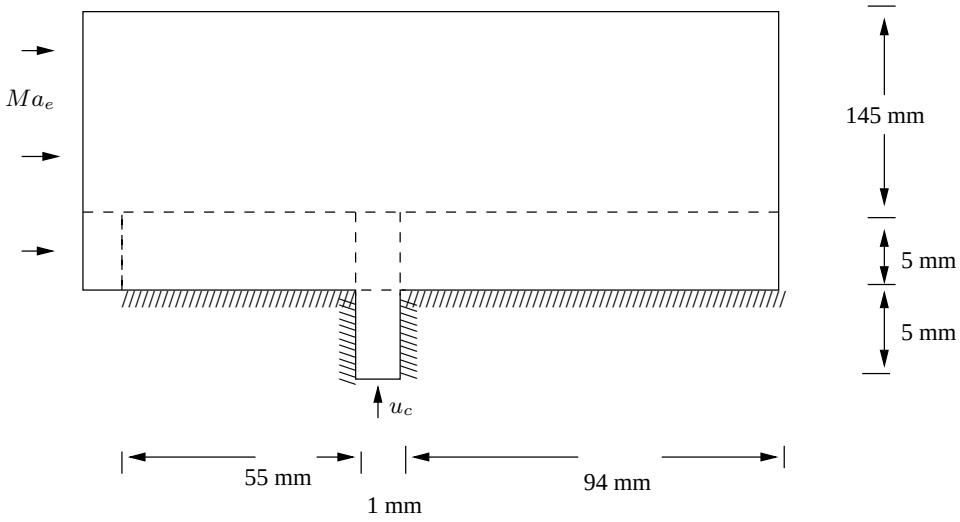


Figure 6: Computational setup: Plate with slot where the dashed lines indicate the boundaries of the grid blocks.

		cond. 1	cond. 2	cond. 3	cond. 4	cond. 5
slot width	[mm]	0.5	1	1	1	1
injection angle	[degrees]	90	90	30	90	60
blowing ratio	[-]	0.065	0.0151	0.065	0.065	0.065
eff. mass flux	[mm]	0.0325	0.0151	0.0325	0.065	0.0563

Table 1: List of injection parameters for different configurations.

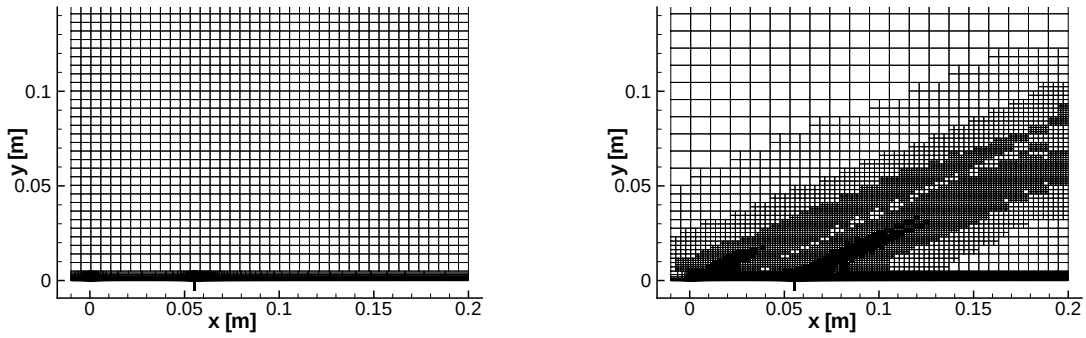
hence, two-dimensional. In order to reduce the computational complexity, we therefore simplify the 3D experimental setup by a two-dimensional configuration of a flat plate (see Fig. 6, where we impose inflow boundary conditions determined by the flow conditions behind the oblique shock. These post-shock conditions correspond to the free stream condition specified in the previous Section 3.1. For the injection, we have considered five configurations with varying slot width L_s, injection angle Θ and maximum blowing ratio F as summarized in Table 1. For all cases the cooling fluid is air at a temperature equal to the wall temperature $T_w = 293K$. Note that Condition 1 and 3 only differ in the smaller injection angle. However, the effective slot width $L_s \sin(\theta)$ is the same in both cases. In our computations, the mass flow of injection gas is either prescribed by imposing a top hat profile via boundary conditions at the slot, i.e.,

$$\frac{\partial p}{\partial n} = 0,\ \rho u = F \cdot \rho_e \cdot Ma_e \cdot c_e \cdot cos(\Theta),\ \rho v = F \cdot \rho_e \cdot Ma_e \cdot c_e \cdot sin(\Theta),\ T = T_w, \qquad (4)$$

or we fully simulate the flow in the injection channel. In the following, we refer to the results of the different approaches by either *simulated* or *modeled* injection, respectively.

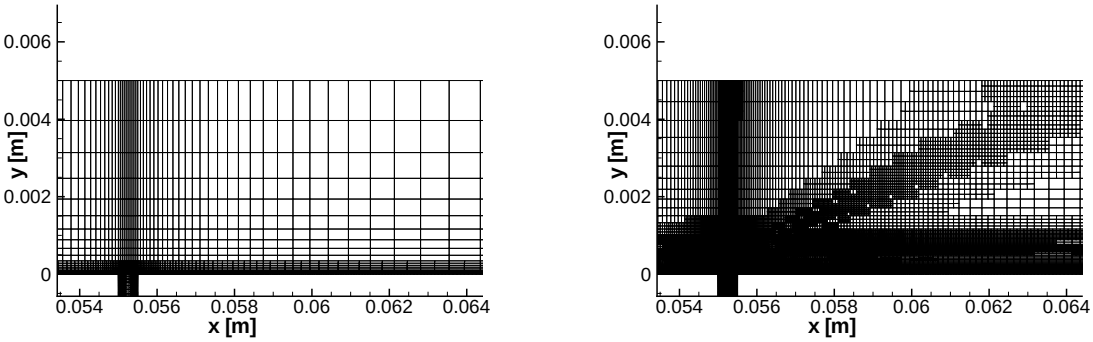
The computations have been performed either on a 5-block (modeled injection) or 6-block (one extra block for the simulated injection) grid, refined by stretching either to the solid wall or, in the other parameter direction, to the leading edge of the plate and to the slot, respectively. The computational domain extends over $\Omega = [-0.01\,m, 0.15\,m] \times [0\,m, 0.15\,m]$. In total the coarse grid consists of 300 grid cells and 4 levels of refinement are used, see Fig. 7.

Numerical results: First of all, we discuss the performance of the grid adaptation where we exemplarily consider the computations corresponding to condition 1 in case of simulated injection. For this configuration, the final adaptive grid corresponding to the steady state solution, consists of about 58000 grid cells. For comparison, the reference computations by Heufer have been performed on a grid for a smaller computational domain that consists of 74000 grid cells. As can be seen in Figs. 7 and 8 the grid has been locally refined in the boundary layer and at the compression waves. Note that there are two isotropic compression waves, one emanating at the leading edge of the plate and another one on top of the injection jet that is shown in Fig. 8(b). The second compression wave results from the interaction of the injection jet with the laminar boundary layer. Due to the injection, the boundary layer becomes thicker and causes a local compression of the attached supersonic flow field and, hence, a compression wave is developing. This is displayed in Fig. 9 for the density field.



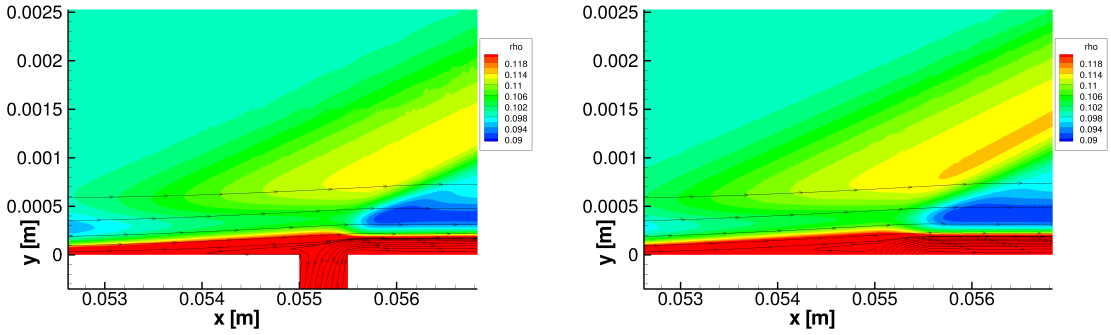
(a) (b)

Figure 7: Condition 1: Initial grid (uniform 1st refinement level) (a) and final grid (b) for simulated injection.



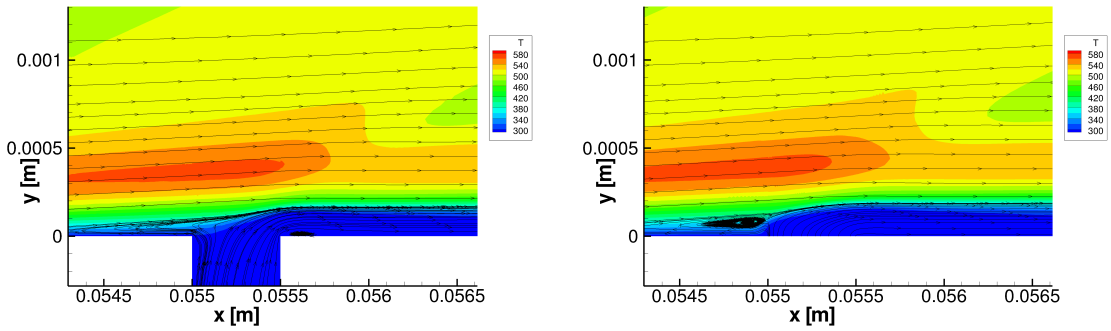
(a) (b)

Figure 8: Condition 1: Zoom into the initial grid (uniform 1st refinement level) (a) and final grid (b) near to the injection slot in case of simulated injection.



(a) (b)

Figure 9: Condition 1: Density distribution and streamlines for simulated (a) and modeled (b) injection.



(a) (b)

Figure 10: Condition 1: Temperature distribution and streamlines for simulated (a) and modeled (b) injection.

Next we discuss the influence of modeling and simulating the injection. For this purpose, we present in Fig. 10 the local temperature fields in the neighborhood of the slot resulting from the numerical simulations corresponding to simulated and modeled injection, respectively. A few millimeter in front of the slot, the main flow deflects from the plate and the boundary layer becomes thicker. Furthermore, there is a separation bubble located in the subsonic area in front of the slot, which arises from a large vortex located there. This vortex seems to be more developed in the case of modeled injection. Additionally, for the simulated injection we observe a flow separation at the upstream wall of the injection slot and another small vortex directly behind the slot.

Although there are visible differences in the flow field, the effect on the cooling efficiency $\eta = 1 - \dot{q}_c/\dot{q}_{nc}$ is small as can be seen in Fig. 11(a). Here the cooling efficiency is defined by the specific wall heat-fluxes $\dot{q}_c$ and $\dot{q}_{nc}$ corresponding to simulations with and without

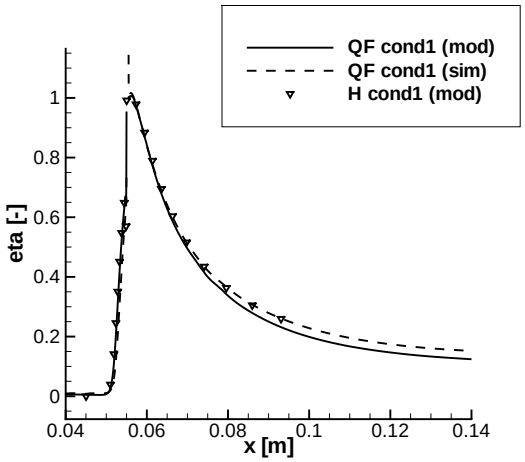
cooling gas injection, respectively. This was already found by Heufer[5]: Far upstream the slot the cooling efficiency is zero, until it starts to increase in the separation bubble. In the injection area it reaches one and then slowly decreases. The only qualitative difference for the simulated injection can be found directly behind the slot, where a cooling efficiency larger than 1 appears. This results from the expanding flow around the corner, where the temperature next to the wall becomes smaller than the wall temperature at this position and the wall heat flux changes its sign. Furthermore, the cooling efficiency seems to decrease faster in the case of modeled injection, but this phenomenon was not observed for other conditions.

For validation of our results, we compare them with experimental measurements performed at the Shock Wave Laboratory at RWTH Aachen and computations with a non-adaptive code, performed by Heufer[4]. As can be concluded from Figs. 11(a) and 11(b), the results for the cooling efficiency are in good agreement and, hence, confirm our computations.

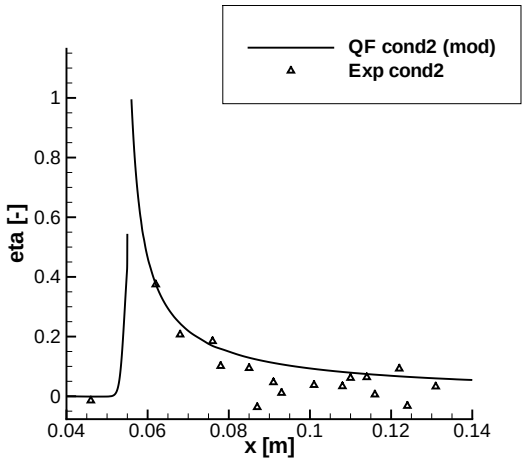
Additionally, we perform parameter studies for varying injection parameters. As can be seen in Fig. 11(d), the lower blowing ratio (condition 2 compared to 4) results in a much faster decreasing cooling efficiency. Decreasing the slot width (condition 1 compared to 4) leads to the same effect. If a critical blowing ratio is reached, the cooling efficiency suddenly becomes negative, because the high mass flux results in the transition of the boundary layer from laminar to turbulent. The used blowing ratio here are all below the critical ones found by Heufer[5].

So far, the direction of injection was orthogonal to the flow field. Now we discuss the influence of injecting the cooling at a different angle, where we confine to the approach of modeled injection. The results for the cooling efficiency are displayed in Fig. 11(c). Note that the injection angle does not change the mass flux. Nevertheless, it influences the cooling efficiency, because the effective slot width changes by the factor of $sin(\Theta)$. Therefore condition 1 and 3 lead to almost the same results, because the factor 2 in the slot widths is compensated by the factor $sin(30°) = 1/2$. Keeping the other parameters fixed and only decreasing the injection angle from 90° (cond. 4) over 60° (cond. 5) to 30° (cond. 3) results in a decreasing cooling efficiency, because of the decreasing effective slot width.

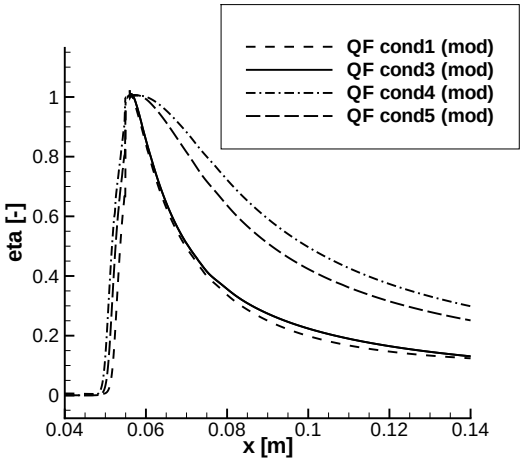
In order to see the quality of the different setups, we finally scale the x-axis with the different effective mass-fluxes $F \cdot L_s \cdot \sin(\Theta)$ (see Table 1). This allows for comparing the cooling efficiencies independently from the mass-flux. As we can conclude from Fig. 12, the results seem to be more or less independent from the geometric parameters. This at least holds for the considered parameter range. There only remains a significant dependency on the effective mass-flux itself, that means, that the cooling efficiency in this regions scales nonlinear with the effective mass-flux.



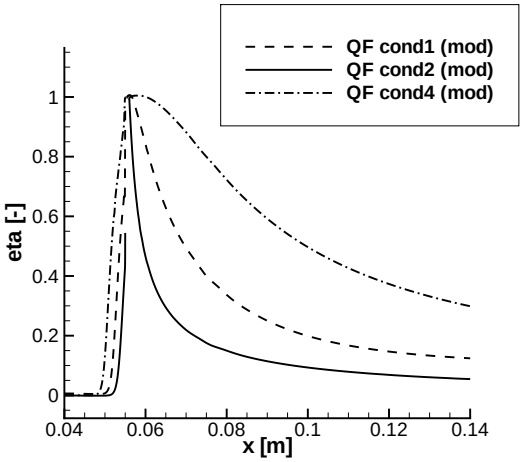
(a) Condition 1: Comparison of modeled (mod) and simulated (sim) injection and reference solution of Heufer[5].



(b) Condition 2: Comparison of modeled (mod) injection and experimental results of SWL.



(c) Comparison of different injection angles.



(d) Comparison of different slot width (cond. 1, 4) and blowing ratio (cond. 2, 4).

Figure 11: Cooling efficiency η

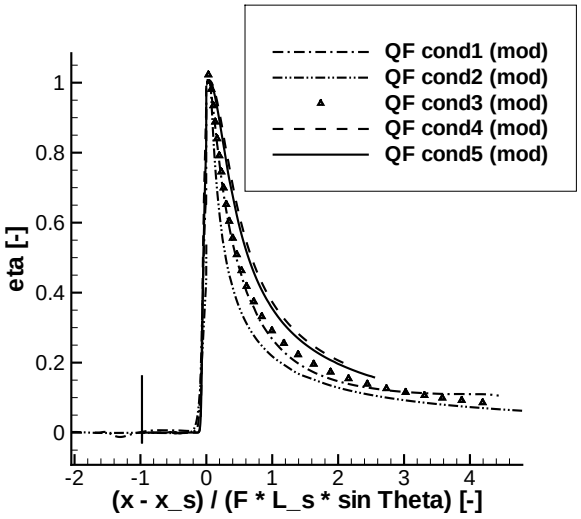


Figure 12: Cooling efficiency η over x normalized by effective mass-flux.

4 Conclusion and Outlook

The numerical investigations confirm that the Quadflow solver is an appropriate tool for simulating film cooling. In particular, the multiscale-based grid adaptation concept turns out to be a reliable tool by which all physically relevant effects are automatically detected and appropriately resolved. Since the grid is only refined in regions, where the flow field exhibits local variations, this results in substantial savings of cells in comparison to a fully refined grid and, hence, the computational costs both in terms of CPU time and memory savings are significantly reduced, whereas the accuracy of the uniform mesh corresponding to the highest refinement level is maintained.

Next we will perform 3D computations, where the cooling gas is injected through a slot of finite length in the plate. In order to reduce the computational cost to an affordable order of magnitude, grid adaptation techniques have to be complemented by parallelization. In recent years, the Quadflow solver has been parallelized. In a first step, the unstructured finite volume solver was parallelized via MPI for distributed memory architectures. Here load-balancing is performed by graph partitioning techniques. However, when applied to adaptive, block-structured grids all data had to be transferred to *one* processor. Since this ruins the overall performance, we are currently parallelizing the multiscale-based grid adaptation that is applied to each block separately. In order to balance the load on the different processors and, thus, to minimize interprocessor communication, we use the concept of space-filling curves. Since the underlying adaptive grids are unstructured due to hanging nodes, this task is by no means trivial. Recently, first results have been published for one-block grids in two and three space dimensions[9]. Currently, the strategy is extended to multi-block grids. This requires an additional handling of ghost cells at

block boundaries.

In the future, the fully parallelized solver is to be applied to turbulent flow fields. Since a DNS is not feasible, we have to model the effect of the unresolved fluctuations on the resolved coarse scales. Up to now, turbulent flow fields have been investigated by means of the Reynolds Averaged Navier-Stokes (RANS) method, where the averaged governing equations are solved for the mean variables. For the turbulent closure, classical one- and two-equation models such as Spalart-Allmaras and Menter-SST, respectively, are used. Recently a differential Reynolds stress model, the SSG/LRR-ω model, has been implemented into Quadflow[1]0. Instead of these models, we will use the Variational Multiscale Method (VMS). This method can be considered as an advanced LES. It has already been implemented into Quadflow according to Koobus and Farhat[1]1 and is currently being validated.

Acknowledgments

Financial support has been provided by the German Research Council (Deutsche Forschungsgemeinschaft - DFG) in the framework of the Sonderforschungsbereich Transregio 40.

REFERENCES

[1] Bramkamp, F., Lamby, Ph. and Müller, S., An adaptive multiscale finite volume solver for unsteady and steady state flow computations, *J. Comp. Phys.*, **197(2)**, 460–490 (2004).

[2] Ballmann, J., Flow Modulation and Fluid-Structure-Interaction at Airplane Wings, Notes on Numerical Fluid Mechanics and Multidisciplinary Design, **84**, *Springer Verlag* (2003).

[3] Anderson, J.D., Hypersonic and High Temperature Gas Dynamics. *McGraw-Hill*, New York, 228–259 (1989).

[4] Heufer, K.A., Untersuchungen zur Filmkühlung in supersonischen Strömungen, *Diss. RWTH Aachen* (2008).

[5] Heufer, K.A. and Olivier, H., Experimental and numerical study of cooling gas injection in laminar, supersonic flow, *AIAA Journal*, **46**, 2741–2751 (2008).

[6] Müller, S., Adaptive Multiscale Schemes for Conservation Laws, Lecture Notes on Computational Science and Engineering, **27**, *Springer Verlag* (2003).

[7] Lamby, Ph., Parametric Multi-Block Grid Generation and Application To Adaptive Flow Simulations, *Diss. RWTH Aachen*, `http://darwin.bth.rwth-aachen.de/opus3/volltexte/2007/1999/pdf/Lamby_Philipp.pdf` (2007).

[8] Bramkamp, B., Unstructured h-Adaptive Finite-Volume Schemes for Compressible Viscous Fluid Flow, *Diss. RWTH Aachen*, `http://darwin.bth.rwth-aachen.de/opus3/volltexte/2003/725/03_255.pdf` (2003).

[9] Brix, K., Mogosan, S., Müller, S. and Schieffer, G., Parallelization of Multiscale-Based Grid Adaptation using Space-Filling Curves, *IGPM-Report*, **299**, RWTH Aachen (2009).

[10] Bosco, A., Reinartz, B. and Müller, S., Differential Reynolds stress model and grid adaptation for hypersonic double wedge simulations, Turbulence, Heat and Mass Transfer 6, Hanjalić, K., Nagano, Y. and Jakirlić, S. (eds.), Proceedings of *6th Int. Symposium on Turbulence, Heat and Mass Transfer*, Rome, Italy, 14-18 September 2009, CD Proceedings (2009).

[11] Koobus, B. and Farhat, Ch., A variational multiscale method for the large eddy simulation of compressible turbulent flows on unstructured meshes – application to vortex shedding, *Computer Methods in Applied Mechanics and Engineering*, **193**, 1367–1383 (2004). .