Integration of RANS and LES flow solvers: interface validation

By J. U. Schlüter, S. Shankaran, S. Kim, H. Pitsch AND J. J. Alonso

1. Introduction

The variety of flow problems encountered in complex flow systems - such as in aero-engine gas turbines - requires well adapted flow solvers for different parts of the system in order to predict the flow accurately and efficiently. Currently, many flow solvers are specialized to simulate one part of a flow system effectively, but are either inadequate or too expensive to be applied to a generic problem.

For the example of a gas turbine, in the compressor and the turbine section, the flow solver has to be able to handle the moving blades, model the wall turbulence, and predict the pressure and density distribution properly. This can be done by a flow solver based on the Reynolds-Averaged Navier-Stokes (RANS) approach. On the other hand, the flow in the combustion chamber is governed by large scale turbulence, chemical reactions, and the presence of fuel spray. Experience shows that predictions of such phenomena are strongly improved by the use of Large Eddy Simulations (LES).

While many design problems of a single flow passage can be addressed by separate computations, only the simultaneous computation of all parts can guarantee the proper prediction of multi-component phenomena, such as compressor/combustor instability and combustor/turbine hot-streak migration. Therefore, a promising strategy to perform full aero-thermal simulations of gas-turbine engines is the use of a RANS flow solver for the compressor sections, an LES flow solver for the combustor, and again a RANS flow solver for the turbine section (figure 1).

While it would be possible to use one single flow solver, which switches between different mathematical approaches depending on the flow section, the current choice is to use several separate flow solvers. The reason for that is, that currently a wide variety of validated flow solvers are in use. Merging two or more computer programs into a single code or extending a code to different modeling approaches is tiresome at best and prone to errors.

The usage of entirely separate flow solvers allows, for a given flow problem, to choose the best combination of a variety of existing flow solvers, which have been developed, optimized, and validated separately. Once these have been equipped with a generic interface, it is possible to continue the development of the flow solvers separately without compromising compatibility. The implementation of this interface into several flow solvers allows their modular exchange, which results in a high degree of flexibility.

The implementation of an interface for the simultaneous flow computation using separate flow solvers faces a number of challenges. These can be described as follows:

(*a*) *Establishing a contact between the flow solvers for information exchange:* The first obvious obstacle is to establish a real-time connection between two or more simultaneous running flow solvers over which the information can be exchanged. Most flow solvers are already parallelized using MPI (Message Passing Interface). Here, MPI will be used for peer-to-peer message passing as well.

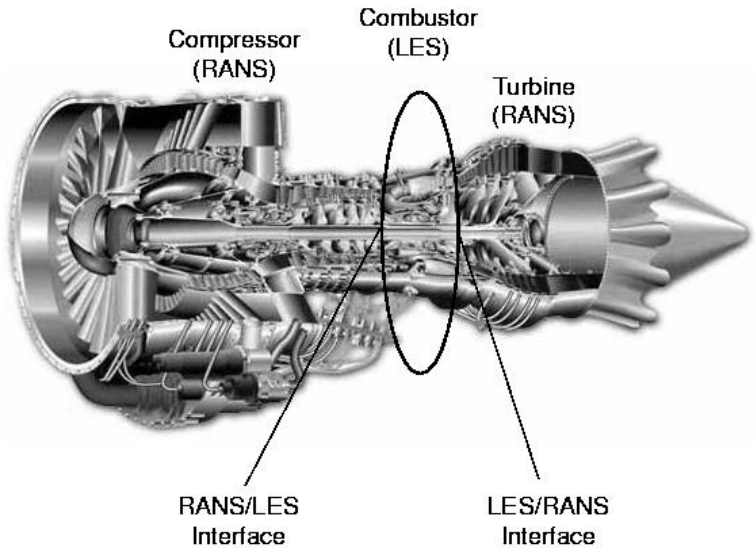

FIGURE 1. Gas turbine engine

(*b*) *Ensuring that each flow solver obtains the information needed on the boundaries:* A general procedure has to be defined, which ensures, that each flow solver knows, which information has to be sent where.

(*c*) *Processing of the obtained information to boundary conditions:* Finally, the physical problem of defining meaningful boundary conditions from the obtained data has to be addressed. This can be especially challenging, when two different modeling approaches, such as LES and RANS, are used.

The current investigation deals with these tasks and describes a successful coupling of RANS and LES flow solvers.

2. Peer-to-peer message passing

The message passing between two separate flow solvers (peer-to-peer message passing) is very similar to the information exchange between processors of a parallel computation. Many flow solvers are parallelized and use MPI for process-to-process message passing. MPI can be used for communication between different flow solvers as well.

Before establishing a contact between two flow solvers, it has to be ensured, that the commands for the internal message passing due to the parallelization of the two codes do not interfere with each other. With MPI it is possible to direct the range of the message passing with communicators. The most commonly used communicator of MPI is the standard communicator `MPI_COMM_WORLD` which includes all processors of all codes. Using this communicator for internal message passing will inevitably result in confusion between the two codes. Hence, each code has to create its own local communicator (*intra*-communicator) to encapsulate the internal message passing. All codes have to use their own intra-communicator for all MPI commands concerning the parallelization of the code instead of `MPI_COMM_WORLD`.

In the next step, a communicator is created for the peer-to-peer message passing (*inter*-communicator). Say, a case with three flow solvers is to be run with a first RANS code using three processors (ranks 0, 1, and 2, local root process 0), a LES code using four

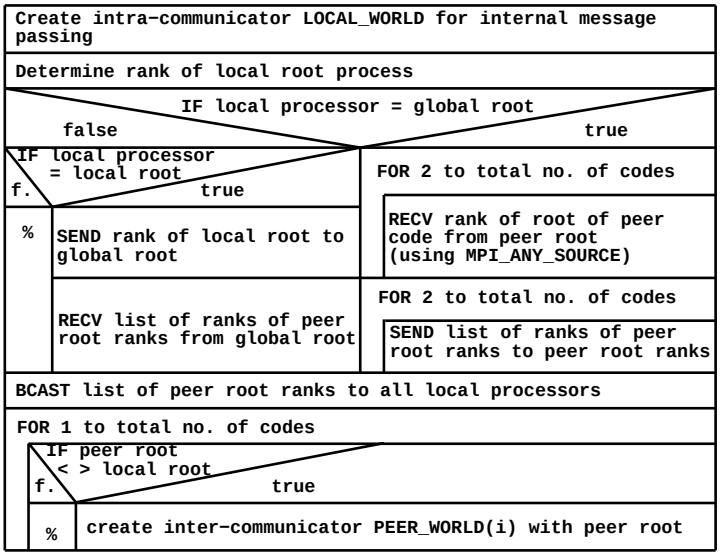

FIGURE 2. Structure Chart for exchange of root ranks needed for creation of inter-communicators

processors (ranks 3, 4, 5, and 6, local root process 3) and a second RANS code using two processors (ranks 7 and 8, local root process 7). In order to create the inter-communicator, it is necessary, that every processor knows the rank of the root processes of the other codes. A global root process is appointed (rank 0) which collects the ranks of the root processes of all codes (here: ranks 0, 3 and 7), compiles them into a list and sends them back to the local root processes. A structure chart for this procedure is shown in figure 2. Since there is no inter-communicator available yet, this communication has to be done with the standard communicator `MPI_COMM_WORLD`. With the knowledge of the ranks of all root processes it is possible to create the inter-communicators.

3. Handshake and communication procedures

3.1. *Handshake*

Efficient parallelizing of a flow solver seeks to limit the information exchange between parallel processes to a minimum, since the information exchange requires a large amount of time compared to the actual computation. Similarly, it is favorable to minimize the communication between several parallel running flow solvers. Since the flow solvers have to exchange flow information rather often, either after each iteration or after a chosen time-step, the aim is to minimize the communication efforts by an initial handshake, which optimizes the communication during the actual flow computation.

The most simple way to organize the information exchange would be to let only the root processes communicate. However, this would mean that prior to the peer-to-peer communication the root processes would have to gather the flow information to hand over from their own processes, and after the peer-to-peer communication would have to broadcast the obtained information back to their processes. The here reported solution avoids this additional communication by direct communication of the neighboring processors on the interface.

The initial handshake routine establishes the direct communication (figure 3). First, for

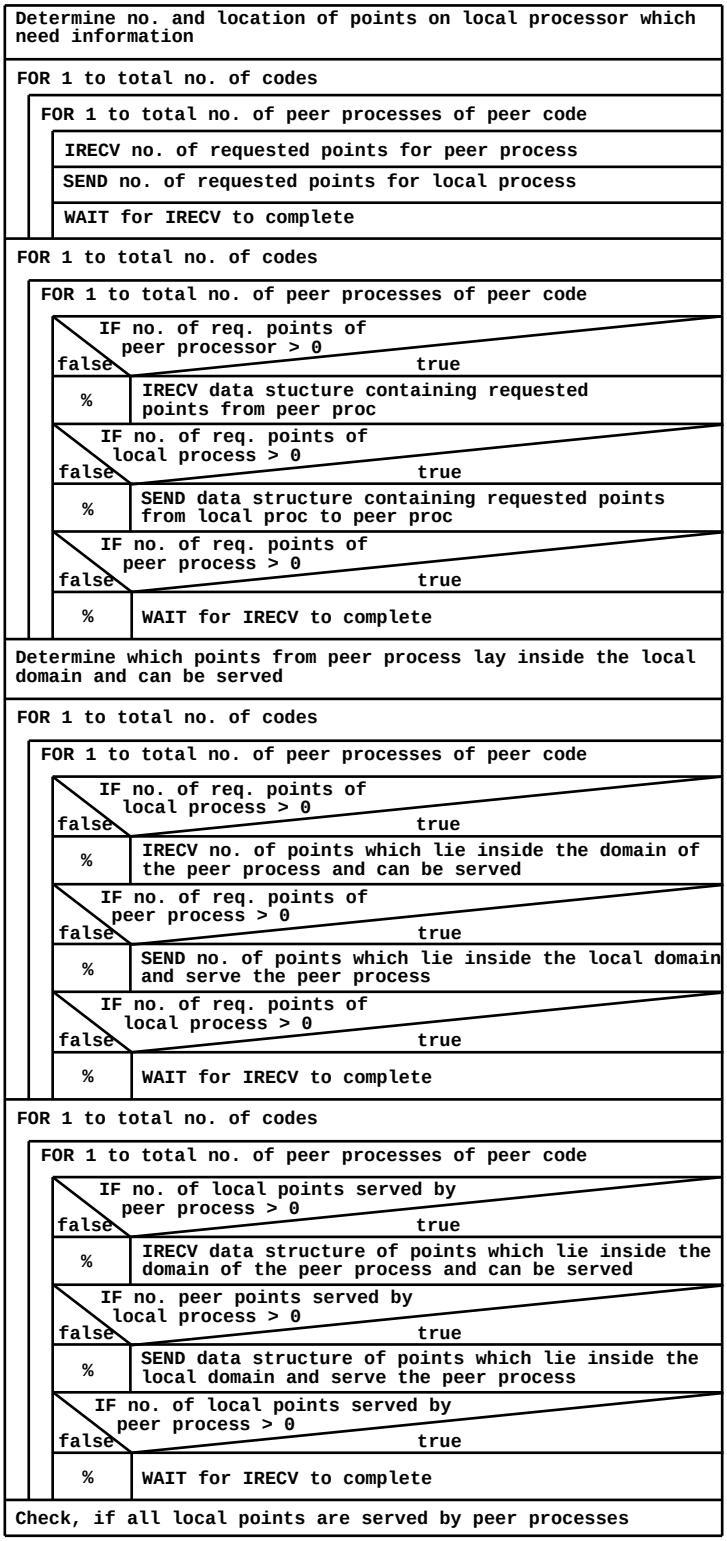

FIGURE 3. Structure Chart for the initial handshake to establish direct communication between interface processors.

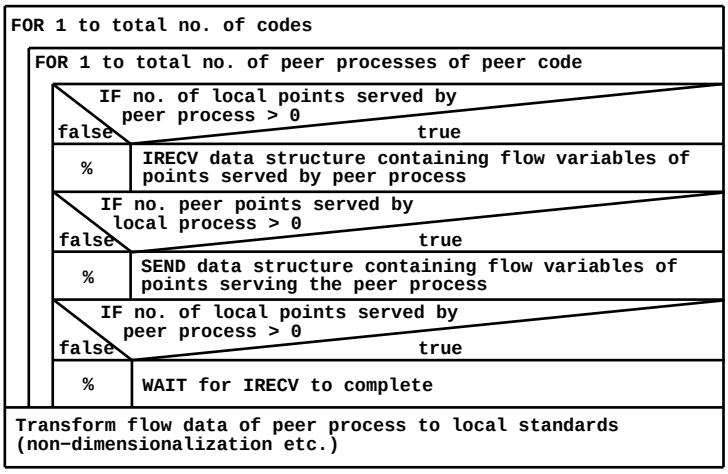

FIGURE 4. Structure Chart for the communication of flow data between iterations.

each code each processor has to identify all the points, which need flow information from the peers to define its interface boundary condition. The location of each of these points has to be stored in a data structure containing three integers and three double precisions. The three integers are an *'ip'* number, which determines what kind of flow variables are requested for this point, an *'id'* number, which contains a unique identification number for each point, and a *'flow solver'* number denoting the flow solver requesting this point. The three real numbers contain the x, y, z-coordinates of the point in Cartesian coordinates using metric dimensions.

The handshake takes place in four steps. First, each processor communicates the number of points in its own domain requesting flow data to each processor of a peer code. This allows each code to dynamically allocate arrays to store received information. In the second step each processor receives a data package containing the location of the requested points from each peer processor that request a non-zero number of points.

In an intermediate step, each processor identifies, whether a requested point lies within its own domain and can be served. During the identification, the interpolation schemes required to obtain the data for this point are also being determined and stored for later use.

In the third communication step each processor communicates to all peer-processes requesting data the number of points found. This allows again to dynamically allocate arrays for the following fourth step. In the fourth communication step, each processor sends out an array to each peer processor it can serve. The array consists of two integers containing ip and id of the point. Finally, each processor determines whether all of its requested point can be served by peer processors.

3.2. *Communication*

The communication of flow data between iterations is rather straight-forward once the handshake is completed (figure 4). Since it is known to every processor what kind of data has to be provided to which peer processor, and from which peer processor flow data can be expected, the data packages can be sent directly without going through the root process.

Each processor has to compile the data to be sent into a data structure. This data structure may vary between different flow solvers and has to be defined beforehand.

However, as a standard data structure a set of 7 variables has been established. These variables contain $\rho, \rho u, \rho v, \rho w, \rho E, k$, and ν_t in this order, with ρ being the density, u, v, w the velocity components in $x-, y-, z-$direction, respectively, E the total Energy, k the turbulent kinetic energy, and ν_t the eddy viscosity. This set of variables gives the freedom to choose between several RANS turbulence models without changing the interface routines, e.g. boundary conditions can be defined from this set of data for $k-\epsilon$ and $k-\omega$ turbulence models likewise. More sophisticated data sets are possible steered by the data structure sent in the handshake routine.

4. Boundary conditions

4.1. *LES Boundary conditions*

The formulation of unsteady LES boundary conditions from ensemble-averaged RANS data is one of the biggest challenges in the coupling of two flow solvers based on such different mathematical approaches like LES and RANS. Unsteady LES boundary conditions have to be generated which fulfill the statistical properties of the time-averaged solution delivered by the RANS flow solver. Even if an unsteady RANS computation is assumed, the time-step of the unsteady RANS computation is usually larger than the LES time-step by several orders of magnitude. The LES boundary conditions then have to correspond to the ensemble-averages delivered by the RANS computation.

4.1.1. *LES inflow boundary conditions*

Specifying inflow conditions for LES from upstream RANS data is a similar problem as specifying LES inflow conditions from experimental data, which is usually given in time-averaged form, and has therefore been investigated in some detail in the past. A method that has been successfully applied is to generate a time-dependent database for the inflow velocity fields by performing a separate LES simulation, in which virtual body forces are applied to achieve the required time-averaged solution (Pierce & Moin, 1998b). However, since unsteady RANS flow solvers may deliver *unsteady* ensemble-averaged velocity profiles, a generation of such a data-base is impossible, since the mean velocity field at the inlet is unknown.

The here proposed LES inlet conditions use a data-base created by a separate LES computation and modifies then its statistical properties in order to match the RANS solution:

$$u_{i,\mathrm{LES}}(t) = \underbrace{\bar{u}_{i,\mathrm{RANS}}}_{I} + \underbrace{(u_{i,\mathrm{DB}}(t) - \bar{u}_{i,\mathrm{DB}})}_{II} \cdot \underbrace{\frac{\sqrt{\overline{u'^2_{(i)}}_{\mathrm{RANS}}}}{\sqrt{\overline{u'^2_{(i)}}_{\mathrm{DB}}}}}_{III} \tag{4.1}$$

with RANS denoting the solution delivered by the RANS computation and DB properties delivered by the database. Term II computes the velocity fluctuation of the database, while term III scales the fluctuation to the actual value needed. When added to term I a meaningful unsteady inlet condition is recovered. In order to keep corrections small, the generated inflow data-base should have statistical properties close to the actual prediction by the RANS flow solver, although it has been shown, that even very generic data-bases are able to recover meaningful LES inflow conditions (Schlüter, 2002).

4.1.2. *LES outflow boundary conditions*

In order to take upstream effects of the downstream flow development into account, LES outflow conditions have to be defined that can impose mean flow properties on the unsteady LES solution matching the statistical properties delivered by a downstream RANS computation. A method, that has been used in the past, employs virtual body forces to drive the mean velocity field of the LES solution to a RANS target velocity field (Schlüter and Pitsch, 2001, Schlüter *et al*, 2002).

$$F_i(\boldsymbol{x}) = \frac{1}{\tau_{\mathrm{F}}}(\bar{u}_{i,\mathrm{RANS}}(\boldsymbol{x}) - \bar{u}_{i,\mathrm{LES}}(\boldsymbol{x}))\,, \tag{4.2}$$

with $\bar{u}_{i,\mathrm{RANS}}$ being the solution of the RANS flow solver computed in an overlap region between LES and RANS domain, and $\bar{u}_{i,\mathrm{LES}}$ is a time-average of the LES solution over a trailing time-window. This body force ensures that the velocity profiles at the outlet of the LES domain fulfill the same statistical properties as the velocity profiles in an overlap region computed by a RANS simulation downstream. This makes it possible to take upstream effects of the downstream flow into account.

4.2. *RANS boundary conditions*

The specification of RANS boundary conditions from LES data is essentially straightforward. The unsteady LES flow data is time-averaged over the time-step applied by the RANS flow solver and can be employed directly as a boundary condition.

In the current study, the compressible formulation of the RANS flow solver and the quasi-incompressible low-Mach-number formulation of the LES code posed a challenge. While the RANS code allows for acoustic waves to propagate in the limits of the RANS formulation and its turbulence models, the density field of the LES solution is entirely defined by chemical reactions and not by acoustics. This leads to the necessity of the RANS inflow and outflow condition to be able to fluctuate the density field at the boundaries in order to let acoustic waves leave the domain.

Currently, the mass-flux vector at every point of the inlet is being specified corresponding to the value delivered by the LES computation. This means ρu, ρv, ρw, (and T) are imposed at the boundaries. This allows the density ρ to fluctuate to account for the passing of acoustic waves. The velocity components u, v, w are adjusted accordingly in order to conserve the mass-flux. Variations of ρ are in the order of $< 2\%$.

Other boundary conditions are possible, especially Navier-Stokes characteristic boundary conditions (Poinsot and Lele, 1992) which have a more accurate treatment of acoustic waves.

5. Validation of the interface

In order to validate the interface and the boundary conditions, a LES flow solver and a RANS flow solver were equipped with the interface and the newly developed boundary conditions. Integrated flow computations were performed in a LES-LES and LES-RANS environment.

5.1. *The LES flow solver*

The LES flow solver chosen for this work, is a code developed at the Center for Turbulence Research at Stanford by Pierce and Moin (Pierce & Moin, 1998a). The flow solver solves the filtered momentum equations with a low-Mach number assumption on an axi-symmetric structured single-block mesh. A second-order finite-volume scheme on

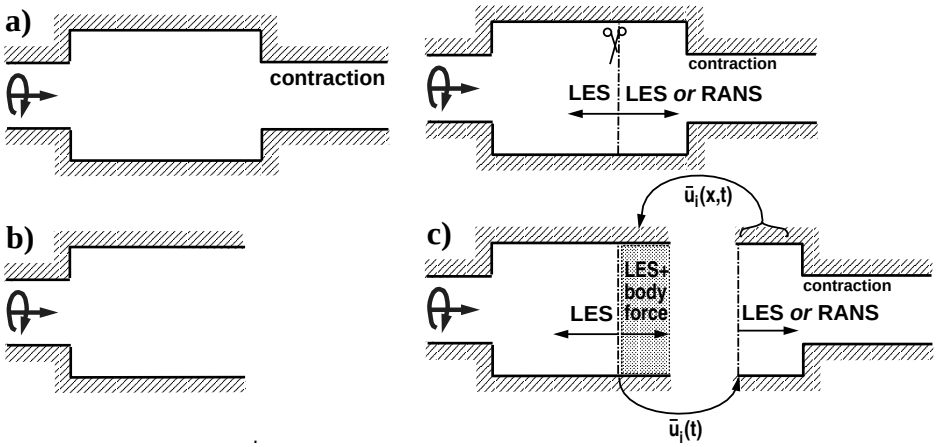

FIGURE 5. Geometry for integrated LES/RANS computations: a) full geometry, b) reduced LES domain, c) schematic splitting of domain to two computational domains

a staggered grid is used (Akselvoll & Moin, 1996). The subgrid stresses are approximated with an eddy-viscosity approach, where the eddy viscosity is determined by a dynamic procedure (Germano *et al*, 1991, Moin *et al*, 1991).

5.2. *The RANS flow Solver*

The RANS flow solver used for this investigation is the TFLO code developed at the Aerospace Computing Lab (ACL) at Stanford. The flow solver computes the unsteady Reynolds Averaged Navier-Stokes equations using a cell-centered discretization on arbitrary multi-block meshes (Yao *et al*, 2000).

The solution procedure is based on efficient explicit modified Runge-Kutta methods with several convergence acceleration techniques such as multi-grid, residual averaging, and local time-stepping. These techniques, multi-grid in particular, provide excellent numerical convergence and fast solution turnaround. Turbulent viscosity is computed from a $k-\omega$ two-equation turbulence model. The dual-time stepping technique (Jameson, 1991, Alonso *et al*, 1995, Belov *et al*, 1996) is used for time-accurate simulations that account for the relative motion of moving parts as well as other sources of flow unsteadiness.

5.3. *Numerical experiment: swirl flow*

The computation of a swirl flow presents a challenging test-case in order to validate the interface and the boundary conditions, due to the complexity of the flow and its sensitivity to inflow and outflow parameters. Yet, this test case is simple enough to perform a LES computation of the entire domain in order to obtain an 'exact' solution, which serves as a reference solution to assess the accuracy of integrated computations.

A swirl flow at an expansion with a subsequent contraction three diameter D downstream of the expansion is considered (figure 5a). Inlet velocity profiles are taken from an actual experiment in a similar geometry (Dellenback *et al*, 1988). The swirl number of the flow is $S = 0.3$, which is just supercritical, meaning that vortex breakdown takes place and a recirculation zone develops. The extension and strength of this recirculation zone is strongly influenced by the presence of the downstream contraction.

In a first computation the entire domain is computed by LES. A first computation of this study computed the entire domain. All subsequent computations assume, that this domain is to be computed by two or more separate flow solvers. The geometry is

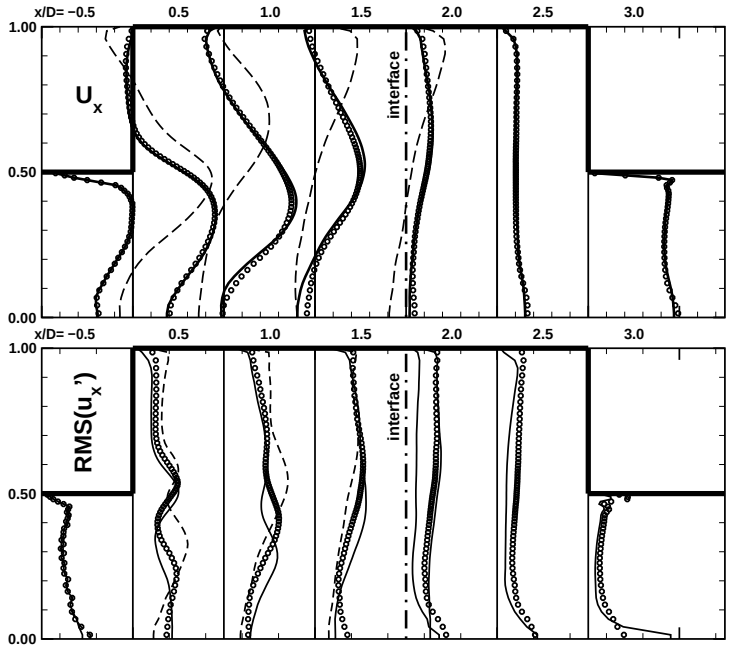

FIGURE 6. Integrated LES/LES computations. Velocity components for different downstream positions. circles: LES of full geometry (figure 5a), dashed line: LES of expansion (figure 5b), solid line: integrated LES-RANS computation (figure 5c)

split in two computational domains with a short overlap region. The expansion is to be computed with the LES code, while the contraction is computed either by a second instance of the LES code or by the RANS code (figure 5c). If the coupling of the two codes is done appropriately, then this coupled simulation should recover the solution of the LES performed for the entire domain.

5.4. *Integrated LES/LES computations*

The first test for the interface and the LES boundary conditions is to use a LES flow solver for the second part of the domain. In these integrated LES/LES computations the same LES flow solver is hence used twice. The time-interval can be chosen arbitrarily when communication between the two instances of the flow solver takes place. Each LES computation can chose a time-step to advance the solution between two iterations of its own, only limited by the CFL condition in its own domain. After several iterations, after both LES computations have computed the same physical time-span, an exchange of time-averaged quantities, the mean velocities $\bar{u}, \bar{v}, \bar{w}$ and the turbulent kinetic energy k, takes place. While it would have been possible in the case of two LES computations to exchange more information, especially about turbulent quantities, it was aim to prove the validity of the LES boundary conditions from section 4.1.

Figure 6 shows the velocity profiles for three different computations. The velocity profiles denoted by the circles represent the LES computation of the entire domain (figure 5a) and hence, the target for the integrated computations.

To show the importance of integrated computations for this case, the dashed lines show the velocity profiles of a LES computation of the expansion, without the computation of the contraction by a second flow solver (figure 5b). It can be seen, that the obtained

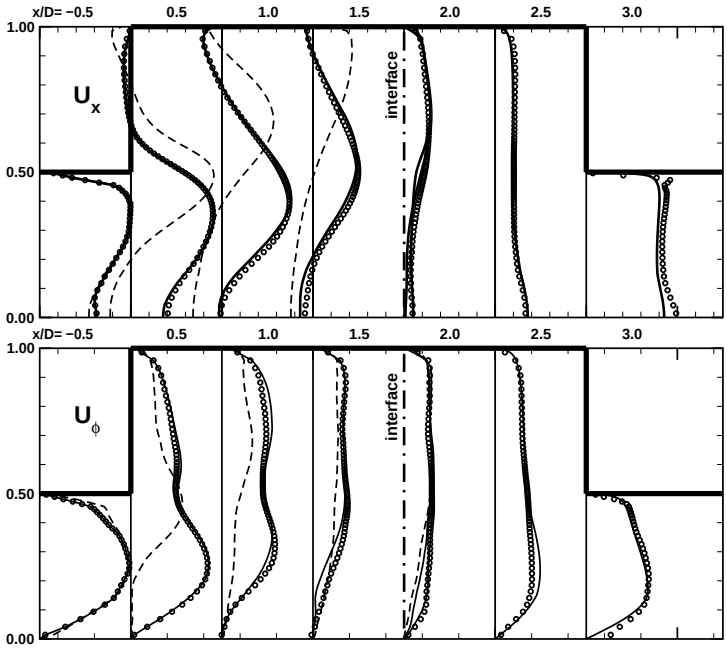

FIGURE 7. Integrated LES/RANS computations. Velocity components for different downstream positions. circles: LES of full geometry (figure 5a), dashed line: LES of expansion (figure 5b), solid line: integrated LES-RANS computation (figure 5c)

velocity field differs substantially from the first simulation, and hence, the influence of the downstream contraction can not be neglected.

The solid lines in figure 6 show the integrated LES/LES-computation using two LES solvers for the two domains (figure 5c). The location of the interface is denoted with a dot-dashed line, meaning, that the velocity profiles on the left-hand side of the interface are computed with the first LES computation and the profiles on the right hand-side from the second LES. The LES computation of the subsequent contraction delivers a mean flow field which is used to correct the outflow conditions of the upstream LES. As a result, the velocity profiles of the integrated LES/LES-computation tend towards the velocity profiles of the LES of the entire domain. The inlet conditions of the second LES are defined from the mean velocity profiles obtained from the upstream LES.

In the integrated LES/LES computation, the velocity fluctuations u'^2, v'^2, w'^2 are handed over as the turbulent kinetic energy $k = 0.5 \cdot (u'^2 + v'^2 + w'^2)$, and reconstructed as $u_i'^2 = 2k/3$. This explains a mismatch in the axial velocity fluctuations at the interface. Although it would have been possible in integrated LES/LES computations to hand over the entire Reynolds-stress tensor, the usage of the RANS standard data set allows better comparison with the following computations.

5.5. *Integrated LES/RANS computations*

The final step in assessing integrated flow computations is to perform a simulation, where the second LES flow solver is replaced by a RANS flow solver. The swirl flow at the expansion is computed by the LES flow solver while the contraction is computed with the RANS flow solver TFLO.

Figure 7 shows the mean velocity profiles obtained by an integrated LES-RANS com-

putation. The circles show the LES of the entire domain, and, for comparison, the dashed line represents the LES solution of the swirl flow without the computation of the contraction.

The integrated LES-RANS computation (solid lines) essentially matches the velocity profiles from the LES of the entire domain. This means, that integrated LES-RANS computations are able to successfully predict complex flows such as the swirl flow considered here.

The time advantage of integrated LES-RANS computations is strongly dependent on the chosen RANS time-step. For the present case, the RANS time step was chosen approximately $2 \cdot 10^3$ times longer than the LES time-step limited by the CFL condition. This resulted in a decrease of computational costs by a factor of ≈ 2.

6. Conclusions

The increasing complexity of flow problems investigated with numerical methods calls for the integration of existing flow solvers, where each of the flow solvers is optimized to address a particular problem. In this study, an interface was developed and implemented that enables two or more flow solvers to run simultaneously and to exchange data at the overlapping boundaries.

The interface was tested on a swirl flow at an expansion with a subsequent contraction, which has been split into two parts, the upstream expansion and the downstream contraction part. Each of these is computed by a separate flow solver in a fully coupled simulation. The integrated LES-LES and LES-RANS computations have demonstrated to yield the same flow prediction as a LES computation of the entire domain.

The LES boundary conditions developed in earlier work were put to a real-time test. RANS boundary conditions were adapted to accommodate the different approaches (compressible/low-Mach number) on both sides of the interface.

The computation reported in this work proves the feasibility, accuracy and efficiency of integrated LES/RANS computations. LES and RANS flow solvers were successfully combined in order to improve the efficiency of the flow prediction without compromising the accuracy. This is an important step towards the application of this concept to industrial applications.

7. Acknowledgments

We gratefully acknowledge support by the US Department of Energy under the ASCI program.

REFERENCES

Akselvoll, K., & Moin, P. 1996 Large-eddy simulation of turbulent confined coannular jets. *J. Fluid Mech.* **315**, 387–411.

J. J. Alonso, L. Martinelli, and A. Jameson 1995 Multigrid unsteady Navier-Stokes calculations with aeroelastic applications. *AIAA Paper* 95-0048.

A. Belov, L. Martinelli, and A. Jameson 1996 Three-dimensional computations of time-dependent incompressible flows with an implicit multigrid-driven algorithm on parallel computers. In *Proc. 15th Int. Conf. on Num. Methods in Fluid Dyn.*

Dellenback, P. A., Metzger, D. E. & Neitzel, G. P 1988 Measurements in

turbulent swirling flow through an abrupt axisymmetric expansion. *AIAA J.* **26**, 669-681.

GERMANO, M., PIOMELLI, U., MOIN, P. & CABOT, W., 1991 A dynamic subgrid-scale eddy viscosity model. *Phys. Fluids* A **(3)**, 1760-1765.

JAMESON, A. 1991 Time dependent calculations using multigrid, with applications to unsteady flows past airfoils and wings. *AIAA paper* 91-1596.

MOIN, P., SQUIRES, K., CABOT, W. & LEE, S. 1991 A dynamic subgrid-scale model for compressible turbulence and scalar transport. *Phys. Fluids* A **(3)**, 2746-2757.

PIERCE, C. D. & MOIN, P. 1998a Large eddy simulation of a confined coaxial jet with swirl and heat release. *AIAA Paper* 98-2892.

PIERCE, C. D. & MOIN, P. 1998b Method for generating equilibrium swirling inflow conditions. *AIAA J.* **36**, 1325-1327.

T. J. POINSOT AND S. K. LELE 1992 Boundary conditions for direct simulations of compressible viscous reacting flows. *J. Comp. Phys.* (101):104–129

J. SCHLÜTER AND H. PITSCH 2001 Consistent boundary conditions for integrated LES/RANS simulations: LES outflow conditions. *Annual Research Briefs* Center for Turbulence Research, NASA Ames/Stanford Univ., 19–30.

J. U. SCHLÜTER, H. PITSCH, AND P. MOIN 2002 Consistent boundary conditions for integrated LES/RANS simulations: LES outflow conditions. *AIAA paper* 2002-3121

J. SCHLÜTER 2002 Consistent boundary conditions for integrated LES/RANS simulations: LES inflow conditions. *Annual Research Briefs* Center for Turbulence Research, NASA Ames/Stanford Univ.

S. SHANKARAN, M.-F. LIOU, N.-S. LIU, AND R. DAVIS ANF J. J. ALONSO 2001 A multi-code-coupling interface for combustor/turbomachinery simulations. *AIAA paper* 2001-0974.

D. VEYNANTE AND T. POINSOT 1996 Reynolds averaged and large eddy simulation modeling for turbulent combustion. In *New Tools in Turbulence Modelling*, Les edition physique. Springer, pp 105–140.

J. YAO, A. JAMESON, J. J. ALONSO, AND F. LIU 2000 Development and validation of a massively parallel flow solver for turbomachinery flows. *AIAA paper* 2000-0882.